Fortify Standalone Report Generator

Developer Workbook

cwe601

Table of Contents

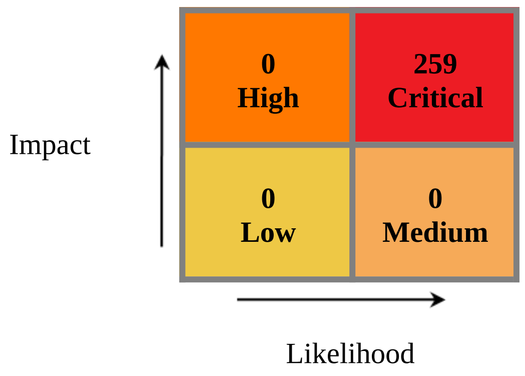
Executive Summary

This workbook is intended to provide all necessary details and information for a developer to understand and remediate the different issues discovered during the cwe601 project audit. The information contained in this workbook is targeted at project managers and developers.

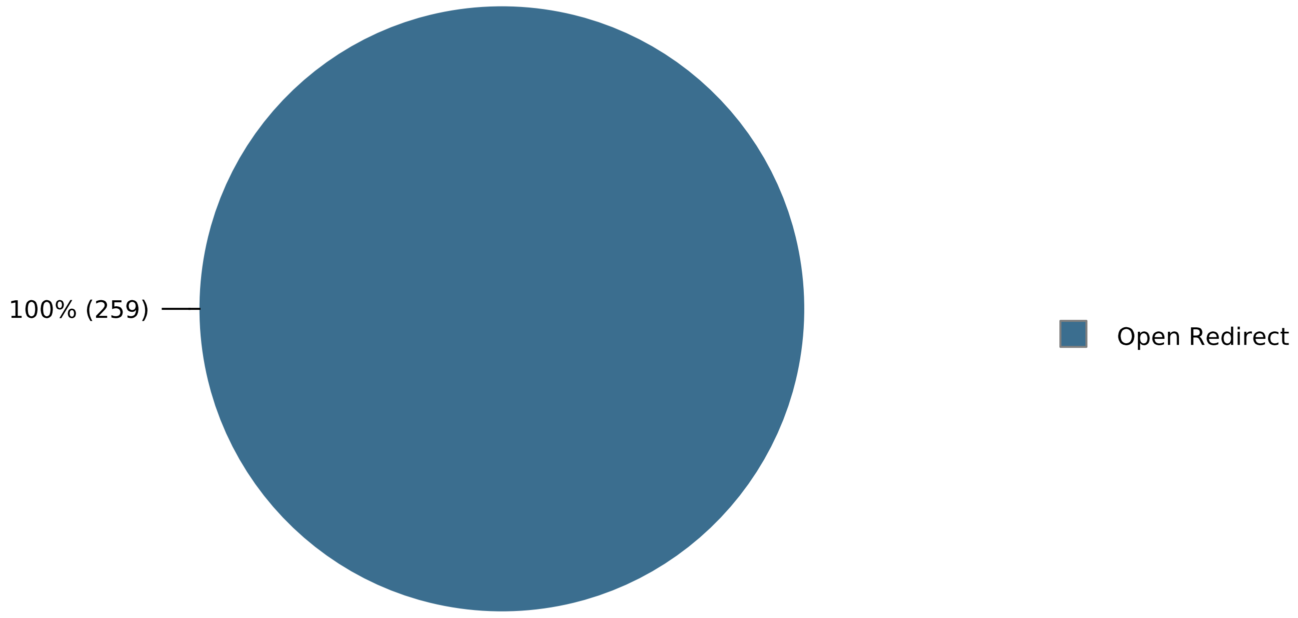
This section provides an overview of the issues uncovered during analysis.

Project Name:	cwe601
Project Version:	
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:	Aug 5, 2021, 9:02 AM	**Engine Version:**	21.1.1.0009
Host Name:	ip-10-138-53-201	**Certification:**	VALID
Number of Files:	552	**Lines of Code:**	32,367

Rulepack Name	Rulepack Version
Fortify Secure Coding Rules, Core, Annotations	2020.4.0.0007
Fortify Secure Coding Rules, Extended, Content	2020.4.0.0007
Fortify Secure Coding Rules, Extended, Java	2020.4.0.0007
Fortify Secure Coding Rules, Extended, JSP	2020.4.0.0007
Fortify Secure Coding Rules, Extended, Configuration	2020.4.0.0007
Fortify Secure Coding Rules, Core, Java	2020.4.0.0007
Fortify Secure Coding Rules, Core, Android	2020.4.0.0007

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	
Open Redirect	0 / 259	0	0	0	0 / 259

Results Outline

Open Redirect (259 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 them 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 victim clicks the link, the code in `Example 1` 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.

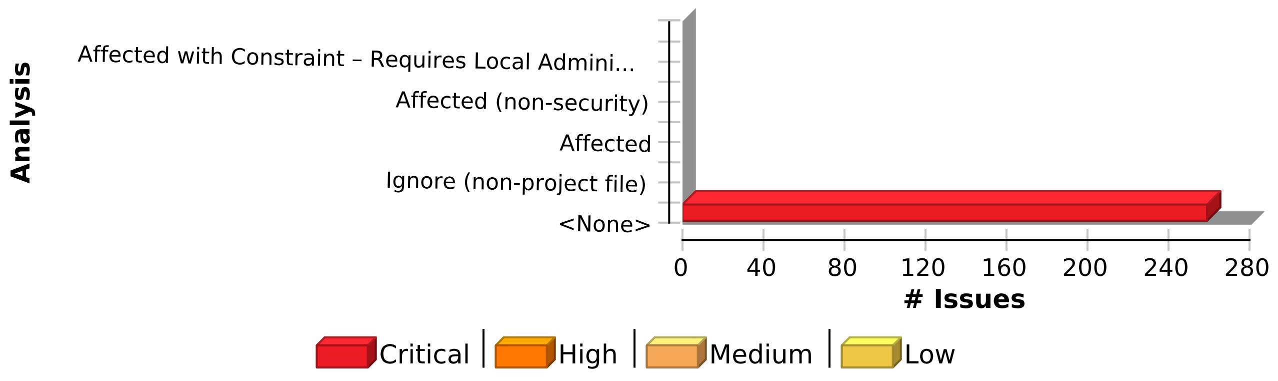
Recommendation

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	259	0	0	259
Total	**259**	**0**	**0**	**259**

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_08.java, line 154 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_08.ba
d
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_08.ja
va:68

```
65 listener = new ServerSocket(39543);
66 socket = listener.accept();
67 /* read input from socket */
68 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
69 readerBuffered = new BufferedReader(readerInputStream);
70 /* POTENTIAL FLAW: Read data using a listening tcp connection */
71 data = readerBuffered.readLine();
```

Sink Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_08.java, line 154 (Open Redirect) — Critical

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_08.java:154
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
151 return;
152 }
153 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
154 response.sendRedirect(data);
155 return;
156 }
157
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getParameter_Servlet_04.java, line 67 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser vlet_04.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser vlet_04.java:42

```
39 if (PRIVATE_STATIC_FINAL_TRUE)
40 {
41 /* POTENTIAL FLAW: Read data from a querystring using getParameter */
42 data = request.getParameter("name");
43 }
44 else
45 {
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_04.java:67
Taint Flags: WEB, XSS

```
64 return;
65 }
66 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
67 response.sendRedirect(data);
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_04.java, line 67 (Open Redirect) — Critical

```
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_73b.java, line 49 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser vlet_73a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser vlet_73a.java:32

```
String data;

/* POTENTIAL FLAW: Read data from a querystring using getParameter */
data = request.getParameter("name");

LinkedList<String> dataLinkedList = new LinkedList<String>();
dataLinkedList.add(0, data);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_73b.java:49
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_05.java, line 119 (Open Redirect) — Critical

Issue Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_05.java, line 119 (Open Redirect) — Critical

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_05
.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_05
.java:56

```
53 InputStreamReader readerInputStream = null;
54 try
55 {
56 readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
57 readerBuffered = new BufferedReader(readerInputStream);
58 /* POTENTIAL FLAW: Read data from a web server with URLConnection */
59 /* This will be reading the first "line" of the response body,
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_05.java:119
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
116 return;
117 }
118 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
119 response.sendRedirect(data);
120 return;
121 }
122
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_45.java, line 59 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_45
.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_45

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_45.java, line 59 (Open Redirect) — Critical

.java:80

```
77
78  try
79  {
80  readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
81  readerBuffered = new BufferedReader(readerInputStream);
82
83  /* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_45.java:59
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
56  return;
57  }
58  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
59  response.sendRedirect(data);
60  return;
61  }
62
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_05.java, line 80 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_Servlet_05.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_05.java:45

```
42  data = ""; /* initialize data in case id is not in query string */
43  /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
44  {
45  StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
46  while (tokenizer.hasMoreTokens())
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_05.java, line 80 (Open Redirect) — Critical

```
{
String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_05.java:80
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_68b.java, line 74 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_68a.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_68a.j
ava:54

```

/* read input from socket */

readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: goodG2BSink()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_68b.java, line 74 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_68b.java:74
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
71  return;
72  }
73  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
74  response.sendRedirect(data);
75  return;
76  }
77
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_72b.java, line 49 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_72a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_72a.java:35

```
32
33  /* Read data from cookies */
34  {
35  Cookie cookieSources[] = request.getCookies();
36  if (cookieSources != null)
37  {
38  /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_72b.java:49
Taint Flags: WEB, XSS

```
46  return;
47  }
48  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
49  response.sendRedirect(data);
50  return;
51  }
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_72b.java, line 49 (Open Redirect) — Critical

```
52
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_41.java, line 53 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_41.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_41.j
ava:78

```
75
76 /* read input from socket */
77
78 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
79 readerBuffered = new BufferedReader(readerInputStream);
80
81 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_41.java:53
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
50 return;
51 }
52 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
53 response.sendRedirect(data);
54 return;
55 }
56
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_67b.java, line 48 (Open Redirect) — Critical

Issue Details

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

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_67b.java, line 48 (Open Redirect) — Critical

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_67
a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_67
a.java:52

```
49
50  try
51  {
52  readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
53  readerBuffered = new BufferedReader(readerInputStream);
54
55  /* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_67b.java:48
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
45  return;
46  }
47  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48  response.sendRedirect(data);
49  return;
50  }
51
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_68b.java, line 74 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_68a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_68a.java:34

```
31
32  /* Read data from cookies */
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_68b.java, line 74 (Open Redirect) — Critical

```
{
Cookie cookieSources[] = request.getCookies();
if (cookieSources != null)
{
/* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: goodG2BSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_68b.java:74
Taint Flags: WEB, XSS

```
71 return;
72 }
73 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
74 response.sendRedirect(data);
75 return;
76 }
77
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_45.java, line 52 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_45.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_45.java:64

```
61 String data;
62
63 /* POTENTIAL FLAW: Read data from a querystring using getParameter */
64 data = request.getParameter("name");
65
66 dataBad = data;
67 badSink(request, response);
```

Sink Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_45.java, line 52 (Open Redirect) — Critical

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_45.java:52
Taint Flags: WEB, XSS

```
49 return;
50 }
51 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
52 response.sendRedirect(data);
53 return;
54 }
55
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_12.java, line 125 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_12.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_12.java
:52

```
49 connection = IO.getDBConnection();
50 /* prepare and execute a (hardcoded) query */
51 preparedStatement = connection.prepareStatement("select name from users where id=0");
52 resultSet = preparedStatement.executeQuery();
53 /* POTENTIAL FLAW: Read data from a database query resultset */
54 data = resultSet.getString(1);
55 }
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_12.java:125
Taint Flags: DATABASE, XSS

```
122 return;
123 }
124 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
125 response.sendRedirect(data);
126 return;
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_12.java, line 125 (Open Redirect) — Critical

```
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_73b.java, line 49 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_73a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_73a.java:48

```

try
{
readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_73b.java:49
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_15.java, line 126 (Open Redirect) — Critical

Issue Details

Kingdom: Input Validation and Representation

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_15.java, line 126 (Open Redirect) — Critical

Scan Engine: SCA (Data Flow)

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_15.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_15.java
:54

```
51 connection = IO.getDBConnection();
52 /* prepare and execute a (hardcoded) query */
53 preparedStatement = connection.prepareStatement("select name from users where id=0");
54 resultSet = preparedStatement.executeQuery();
55 /* POTENTIAL FLAW: Read data from a database query resultset */
56 data = resultSet.getString(1);
57 }
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_15.java:126
Taint Flags: DATABASE, XSS

```
123 return;
124 }
125 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
126 response.sendRedirect(data);
127 return;
128 }
129
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_51b.java, line 46 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_51
a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_51
a.java:46

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_51b.java, line 46 (Open Redirect) — Critical

```
43
44 try
45 {
46 readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
47 readerBuffered = new BufferedReader(readerInputStream);
48
49 /* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_51b.java:46
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
43 return;
44 }
45 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
46 response.sendRedirect(data);
47 return;
48 }
49
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_42.java, line 129 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_42.badSource
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_42.java:53

```
50
51 /* read input from socket */
52
53 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
54 readerBuffered = new BufferedReader(readerInputStream);
55
56 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_42.java, line 129 (Open Redirect) — Critical

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_42.java:129
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
126 return;
127 }
128 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
129 response.sendRedirect(data);
130 return;
131 }
132
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_04.java, line 119 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_04
.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_04
.java:56

```
53 InputStreamReader readerInputStream = null;
54 try
55 {
56 readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
57 readerBuffered = new BufferedReader(readerInputStream);
58 /* POTENTIAL FLAW: Read data from a web server with URLConnection */
59 /* This will be reading the first "line" of the response body,
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_04.java:119
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
116 return;
117 }
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_04.java, line 119 (Open Redirect) — Critical

```
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_10.java, line 140 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_10.ba
d
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_10.ja
va:54

```
listener = new ServerSocket(39543);
socket = listener.accept();
/* read input from socket */
readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
/* POTENTIAL FLAW: Read data using a listening tcp connection */
data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_10.java:140
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_URLConnection_16.java, line 108 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_16
.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_16
.java:50

```
47  InputStreamReader readerInputStream = null;
48  try
49  {
50  readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
51  readerBuffered = new BufferedReader(readerInputStream);
52  /* POTENTIAL FLAW: Read data from a web server with URLConnection */
53  /* This will be reading the first "line" of the response body,
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_16.java:108
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
105  return;
106  }
107  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
108  response.sendRedirect(data);
109  return;
110  }
111
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_URLConnection_03.java, line 112 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_03
.bad

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_URLConnection_03.java, line 112 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_03
.java:49

```
46 InputStreamReader readerInputStream = null;
47 try
48 {
49 readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
50 readerBuffered = new BufferedReader(readerInputStream);
51 /* POTENTIAL FLAW: Read data from a web server with URLConnection */
52 /* This will be reading the first "line" of the response body,
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_03.java:112
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
109 return;
110 }
111 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
112 response.sendRedirect(data);
113 return;
114 }
115
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_connect_tcp_04.java, line 132 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_04.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_04.j
ava:58

```
55 /* Read data using an outbound tcp connection */
56 socket = new Socket("host.example.org", 39544);
57 /* read input from socket */
58 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
59 readerBuffered = new BufferedReader(readerInputStream);
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_04.java, line 132 (Open Redirect) — Critical

```
/* POTENTIAL FLAW: Read data using an outbound tcp connection */
data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_04.java:132
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
129 return;
130 }
131 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
132 response.sendRedirect(data);
133 return;
134 }
135
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_12.java, line 69 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl et_12.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl et_12.java:37

```
34 data = ""; /* initialize data in case there are no cookies */
35 /* Read data from cookies */
36 {
37 Cookie cookieSources[] = request.getCookies();
38 if (cookieSources != null)
39 {
40 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_12.java:69
Taint Flags: WEB, XSS

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_12.java, line 69 (Open Redirect) — Critical

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_03.java, line 140 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_03.ba
d
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_03.ja
va:54

```
listener = new ServerSocket(39543);
socket = listener.accept();
/* read input from socket */
readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
/* POTENTIAL FLAW: Read data using a listening tcp connection */
data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_03.java:140
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_71b.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_71a.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_71a.j
ava:54

```
51
52  /* read input from socket */
53
54  readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
55  readerBuffered = new BufferedReader(readerInputStream);
56
57  /* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_71b.java:48
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
45  return;
46  }
47  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48  response.sendRedirect(data);
49  return;
50  }
51
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_31.java, line 60 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_31.bad

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_31.java, line 60 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_31.java:37

```
String data;

/* POTENTIAL FLAW: Read data from a querystring using getParameter */
data = request.getParameter("name");

dataCopy = data;
}
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_31.java:60
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_81_bad.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_81a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_81a.java:47

```

try
{
readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_URLConnection_81_bad.java, line 47 (Open Redirect) — Critical

```
49
50  /* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: action()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_81_bad.java:47
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
44  return;
45  }
46  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47  response.sendRedirect(data);
48  return;
49  }
50
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_database_71b.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_71a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_71a.jav
a:51

```
48
49  /* prepare and execute a (hardcoded) query */
50  preparedStatement = connection.prepareStatement("select name from users where id=0");
51  resultSet = preparedStatement.executeQuery();
52
53  /* POTENTIAL FLAW: Read data from a database query resultset */
54  data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_71b.java:48
Taint Flags: DATABASE, XSS

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_71b.java, line 48 (Open Redirect) — Critical

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_01.java, line 107 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_01
.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_01
.java:50

```

try
{
readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_01.java:107
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_17.java, line 139 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_17.ba
d
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_17.ja
va:57

```
54
55  /* read input from socket */
56
57  readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
58  readerBuffered = new BufferedReader(readerInputStream);
59
60  /* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_17.java:139
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
136  return;
137  }
138  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
139  response.sendRedirect(data);
140  return;
141  }
142  }
```

**testcases/CWE601_Open_Redirect/
CWE601_Open_Redirect__Servlet_connect_tcp_22a.java, line 55 (Open Redirect)** — **Critical**

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_22b.
badSource

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_22a.java, line 55 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_22b.java:49

```
46 /* Read data using an outbound tcp connection */
47 socket = new Socket("host.example.org", 39544);
48 /* read input from socket */
49 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
50 readerBuffered = new BufferedReader(readerInputStream);
51 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
52 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_22a.java:55
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
52 return;
53 }
54 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
55 response.sendRedirect(data);
56 return;
57 }
58
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_68b.java, line 74 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_Servlet_68a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_68a.java:35

```
32
33 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
34 {
35 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_68b.java, line 74 (Open Redirect) — Critical

```
while (tokenizer.hasMoreTokens())
{
String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: goodG2BSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_68b.java:74
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
71 return;
72 }
73 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
74 response.sendRedirect(data);
75 return;
76 }
77
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_68b.java, line 74 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_68
a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_68
a.java:47

```
44
45 try
46 {
47 readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
48 readerBuffered = new BufferedReader(readerInputStream);
49
50 /* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_68b.java, line 74 (Open Redirect) — Critical

Enclosing Method: goodG2BSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_68b.java:74
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_06.java, line 118 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_06
.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_06
.java:55

```
InputStreamReader readerInputStream = null;
try
{
readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
/* POTENTIAL FLAW: Read data from a web server with URLConnection */
/* This will be reading the first "line" of the response body,
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_06.java:118
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_06.java, line 118 (Open Redirect) — Critical

```
120 }
121
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_14.java, line 125 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_14.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_14.j
ava:51

```
48 /* Read data using an outbound tcp connection */
49 socket = new Socket("host.example.org", 39544);
50 /* read input from socket */
51 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
52 readerBuffered = new BufferedReader(readerInputStream);
53 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
54 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_14.java:125
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
122 return;
123 }
124 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
125 response.sendRedirect(data);
126 return;
127 }
128
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_10.java, line 60 (Open Redirect) — Critical

Issue Details

Kingdom: Input Validation and Representation

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_10.java, line 60 (Open Redirect) — Critical

Scan Engine: SCA (Data Flow)

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_10.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_10.java:35

```
32  if (IO.staticTrue)
33  {
34  /* POTENTIAL FLAW: Read data from a querystring using getParameter */
35  data = request.getParameter("name");
36  }
37  else
38  {
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_10.java:60
Taint Flags: WEB, XSS

```
57  return;
58  }
59  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
60  response.sendRedirect(data);
61  return;
62  }
63
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_01.java, line 61 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_01.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_01.java:37

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_01.java, line 61 (Open Redirect) — **Critical**

```
34
35 /* Read data from cookies */
36 {
37 Cookie cookieSources[] = request.getCookies();
38 if (cookieSources != null)
39 {
40 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_01.java:61
Taint Flags: WEB, XSS

```
58 return;
59 }
60 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
61 response.sendRedirect(data);
62 return;
63 }
64
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_42.java, line 68 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl et_42.badSource
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl et_42.java:36

```
33
34 /* Read data from cookies */
35 {
36 Cookie cookieSources[] = request.getCookies();
37 if (cookieSources != null)
38 {
39 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getCookies_Servlet_42.java, line 68 (Open Redirect) — Critical

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_42.java:68
Taint Flags: WEB, XSS

```
65 return;
66 }
67 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
68 response.sendRedirect(data);
69 return;
70 }
71
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_06.java, line 130 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_06.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_06.java
:58

```
55 connection = IO.getDBConnection();
56 /* prepare and execute a (hardcoded) query */
57 preparedStatement = connection.prepareStatement("select name from users where id=0");
58 resultSet = preparedStatement.executeQuery();
59 /* POTENTIAL FLAW: Read data from a database query resultset */
60 data = resultSet.getString(1);
61 }
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_06.java:130
Taint Flags: DATABASE, XSS

```
127 return;
128 }
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_06.java, line 130 (Open Redirect) — Critical

```
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getCookies_Servlet_41.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl et_41.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl et_41.java:61

```

/* Read data from cookies */
{
Cookie cookieSources[] = request.getCookies();
if (cookieSources != null)
{
/* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_41.java:47
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_45.java, line 53 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_Servlet_45.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_45.java:68

```
65
66  /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
67  {
68  StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
69  while (tokenizer.hasMoreTokens())
70  {
71  String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_45.java:53
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
50  return;
51  }
52  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
53  response.sendRedirect(data);
54  return;
55  }
56
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_11.java, line 112 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_11

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_11.java, line 112 (Open Redirect) — Critical

.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_11
.java:49

```
InputStreamReader readerInputStream = null;
try
{
readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
/* POTENTIAL FLAW: Read data from a web server with URLConnection */
/* This will be reading the first "line" of the response body,
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_11.java:112
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_66b.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_66a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_66a.java:31

```
String data;

/* POTENTIAL FLAW: Read data from a querystring using getParameter */
data = request.getParameter("name");
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_66b.java, line 48 (Open Redirect) — Critical

```
32
33 String[] dataArray = new String[5];
34 dataArray[2] = data;
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_66b.java:48
Taint Flags: WEB, XSS

```
45 return;
46 }
47 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48 response.sendRedirect(data);
49 return;
50 }
51
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_02.java, line 60 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Servlet_02.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_02.java:35

```
32 if (true)
33 {
34 /* POTENTIAL FLAW: Read data from a querystring using getParameter */
35 data = request.getParameter("name");
36 }
37 else
38 {
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_02.java, line 60 (Open Redirect) — **Critical**

Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_02.java:60
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_10.java, line 73 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_10.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_10.java:38

```
data = ""; /* initialize data in case id is not in query string */
/* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
{
StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
while (tokenizer.hasMoreTokens())
{
String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_10.java:73
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_10.java, line 73 (Open Redirect) — Critical

```
74  return;
75  }
76
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_09.java, line 112 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_09
.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_09
.java:49

```
46  InputStreamReader readerInputStream = null;
47  try
48  {
49  readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
50  readerBuffered = new BufferedReader(readerInputStream);
51  /* POTENTIAL FLAW: Read data from a web server with URLConnection */
52  /* This will be reading the first "line" of the response body,
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_09.java:112
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
109  return;
110  }
111  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
112  response.sendRedirect(data);
113  return;
114  }
115
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_07.java, line 74 (Open Redirect) — Critical

Issue Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_07.java, line 74 (Open Redirect) — Critical

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_07.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_07.java:43

```
40 data = ""; /* initialize data in case there are no cookies */
41 /* Read data from cookies */
42 {
43 Cookie cookieSources[] = request.getCookies();
44 if (cookieSources != null)
45 {
46 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_07.java:74
Taint Flags: WEB, XSS

```
71 return;
72 }
73 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
74 response.sendRedirect(data);
75 return;
76 }
77
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_72b.java, line 49 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_72a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_72a.jav
a:52

Open Redirect	Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_database_72b.java, line 49 (Open Redirect)	Critical

```

/* prepare and execute a (hardcoded) query */
preparedStatement = connection.prepareStatement("select name from users where id=0");
resultSet = preparedStatement.executeQuery();

/* POTENTIAL FLAW: Read data from a database query resultset */
data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_72b.java:49
Taint Flags: DATABASE, XSS

```
46 return;
47 }
48 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
49 response.sendRedirect(data);
50 return;
51 }
52
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getParameter_Servlet_14.java, line 60 (Open Redirect)	Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser vlet_14.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser vlet_14.java:35

```
32 if (IO.staticFive == 5)
33 {
34 /* POTENTIAL FLAW: Read data from a querystring using getParameter */
35 data = request.getParameter("name");
36 }
37 else
38 {
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_14.java, line 60 (Open Redirect) — Critical

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_14.java:60
Taint Flags: WEB, XSS

```
57 return;
58 }
59 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
60 response.sendRedirect(data);
61 return;
62 }
63
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_51b.java, line 46 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Servlet_51a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_51a.java:30

```
27 String data;
28
29 /* POTENTIAL FLAW: Read data from a querystring using getParameter */
30 data = request.getParameter("name");
31
32 (new CWE601_Open_Redirect__Servlet_getParameter_Servlet_51b()).badSink(data , request, response );
33 }
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_51b.java:46
Taint Flags: WEB, XSS

```
43 return;
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_51b.java, line 46 (Open Redirect) — Critical

```
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_51b.java, line 46 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_51a.
bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_51a.
java:50

```

/* read input from socket */

readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_51b.java:46
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

Open Redirect | Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_41.java, line 53 (Open Redirect) | Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_41.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_41.java
:78

```
75
76 /* prepare and execute a (hardcoded) query */
77 preparedStatement = connection.prepareStatement("select name from users where id=0");
78 resultSet = preparedStatement.executeQuery();
79
80 /* POTENTIAL FLAW: Read data from a database query resultset */
81 data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_41.java:53
Taint Flags: DATABASE, XSS

```
50 return;
51 }
52 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
53 response.sendRedirect(data);
54 return;
55 }
56
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_74b.java, line 49 (Open Redirect) | Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_74a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_74b.java, line 49 (Open Redirect) — Critical

et_74a.java:35

```
/* Read data from cookies */
{
Cookie cookieSources[] = request.getCookies();
if (cookieSources != null)
{
/* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_74b.java:49
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_17.java, line 63 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_17.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_17.java:37

```
/* Read data from cookies */
{
Cookie cookieSources[] = request.getCookies();
if (cookieSources != null)
{
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getCookies_Servlet_17.java, line 63 (Open Redirect) — Critical

```
40 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_17.java:63
Taint Flags: WEB, XSS

```
60 return;
61 }
62 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
63 response.sendRedirect(data);
64 return;
65 }
66 }
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getParameter_Servlet_22a.java, line 55 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser vlet_22b.badSource
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser vlet_22b.java:33

```
30 if (CWE601_Open_Redirect__Servlet_getParameter_Servlet_22a.badPublicStatic)
31 {
32 /* POTENTIAL FLAW: Read data from a querystring using getParameter */
33 data = request.getParameter("name");
34 }
35 else
36 {
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_22a.java:55
Taint Flags: WEB, XSS

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_22a.java, line 55 (Open Redirect) — Critical

```
52 return;
53 }
54 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
55 response.sendRedirect(data);
56 return;
57 }
58
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_03.java, line 60 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Servlet_03.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_03.java:35

```
32 if (5 == 5)
33 {
34 /* POTENTIAL FLAW: Read data from a querystring using getParameter */
35 data = request.getParameter("name");
36 }
37 else
38 {
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_03.java:60
Taint Flags: WEB, XSS

```
57 return;
58 }
59 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
60 response.sendRedirect(data);
61 return;
62 }
63
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_15.java, line 142 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_15.ba
d
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_15.ja
va:56

```
53 listener = new ServerSocket(39543);
54 socket = listener.accept();
55 /* read input from socket */
56 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
57 readerBuffered = new BufferedReader(readerInputStream);
58 /* POTENTIAL FLAW: Read data using a listening tcp connection */
59 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_15.java:142
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
139 return;
140 }
141 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
142 response.sendRedirect(data);
143 return;
144 }
145
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getParameter_Servlet_13.java, line 60 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_13.bad

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getParameter_Servlet_13.java, line 60 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_13.java:35

```
if (IO.STATIC_FINAL_FIVE == 5)
{
/* POTENTIAL FLAW: Read data from a querystring using getParameter */
data = request.getParameter("name");
}
else
{
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_13.java:60
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_connect_tcp_15.java, line 127 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_15.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_15.java:53

```
/* Read data using an outbound tcp connection */
socket = new Socket("host.example.org", 39544);
/* read input from socket */
readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_15.java, line 127 (Open Redirect) — **Critical**

```
55 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
56 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_15.java:127
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
124 return;
125 }
126 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
127 response.sendRedirect(data);
128 return;
129 }
130
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_06.java, line 66 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Servlet_06.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_06.java:41

```
38 if (PRIVATE_STATIC_FINAL_FIVE == 5)
39 {
40 /* POTENTIAL FLAW: Read data from a querystring using getParameter */
41 data = request.getParameter("name");
42 }
43 else
44 {
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_06.java:66
Taint Flags: WEB, XSS

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_06.java, line 66 (Open Redirect) — Critical

```
63 return;
64 }
65 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
66 response.sendRedirect(data);
67 return;
68 }
69
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_16.java, line 120 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_16.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_16.java
:53

```
50 connection = IO.getDBConnection();
51 /* prepare and execute a (hardcoded) query */
52 preparedStatement = connection.prepareStatement("select name from users where id=0");
53 resultSet = preparedStatement.executeQuery();
54 /* POTENTIAL FLAW: Read data from a database query resultset */
55 data = resultSet.getString(1);
56 }
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_16.java:120
Taint Flags: DATABASE, XSS

```
117 return;
118 }
119 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
120 response.sendRedirect(data);
121 return;
122 }
123
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_09.java, line 73 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S ervlet_09.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S ervlet_09.java:38

```
35 data = ""; /* initialize data in case id is not in query string */
36 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
37 {
38 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
39 while (tokenizer.hasMoreTokens())
40 {
41 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_09.java:73
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
70 return;
71 }
72 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
73 response.sendRedirect(data);
74 return;
75 }
76
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_07.java, line 118 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_07

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_07.java, line 118 (Open Redirect) — Critical

.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_07
.java:55

```
52  InputStreamReader readerInputStream = null;
53  try
54  {
55  readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
56  readerBuffered = new BufferedReader(readerInputStream);
57  /* POTENTIAL FLAW: Read data from a web server with URLConnection */
58  /* This will be reading the first "line" of the response body,
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_07.java:118
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
115  return;
116  }
117  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
118  response.sendRedirect(data);
119  return;
120  }
121
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_66b.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_66a.
bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_66a.
java:51

```
48
49  /* read input from socket */
50
51  readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_66b.java, line 48 (Open Redirect) — Critical

```
52 readerBuffered = new BufferedReader(readerInputStream);
53
54 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_66b.java:48
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
45 return;
46 }
47 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48 response.sendRedirect(data);
49 return;
50 }
51
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_16.java, line 136 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_16.ba
d
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_16.ja
va:55

```
52 listener = new ServerSocket(39543);
53 socket = listener.accept();
54 /* read input from socket */
55 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
56 readerBuffered = new BufferedReader(readerInputStream);
57 /* POTENTIAL FLAW: Read data using a listening tcp connection */
58 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_16.java, line 136 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_16.java:136
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
133 return;
134 }
135 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
136 response.sendRedirect(data);
137 return;
138 }
139
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_68b.java, line 74 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_68a.
bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_68a.
java:51

```
48
49 /* read input from socket */
50
51 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
52 readerBuffered = new BufferedReader(readerInputStream);
53
54 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: goodG2BSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_68b.java:74
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
71 return;
72 }
73 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
74 response.sendRedirect(data);
75 return;
76 }
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_68b.java, line 74 (Open Redirect) — Critical

```
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_22a.java, line 55 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_22b.bad Source
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_22b.java:50

```
connection = IO.getDBConnection();
/* prepare and execute a (hardcoded) query */
preparedStatement = connection.prepareStatement("select name from users where id=0");
resultSet = preparedStatement.executeQuery();
/* POTENTIAL FLAW: Read data from a database query resultset */
data = resultSet.getString(1);
}
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_22a.java:55
Taint Flags: DATABASE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_68b.java, line 74 (Open Redirect) — Critical

Issue Details

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

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_68b.java, line 74 (Open Redirect) — Critical

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_68a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_68a.jav
a:51

```
48
49 /* prepare and execute a (hardcoded) query */
50 preparedStatement = connection.prepareStatement("select name from users where id=0");
51 resultSet = preparedStatement.executeQuery();
52
53 /* POTENTIAL FLAW: Read data from a database query resultset */
54 data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: goodG2BSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_68b.java:74
Taint Flags: DATABASE, XSS

```
71 return;
72 }
73 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
74 response.sendRedirect(data);
75 return;
76 }
77
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_74b.java, line 49 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_74
a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_74
a.java:48

```
45
46 try
47 {
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_74b.java, line 49 (Open Redirect) — Critical

```
48 readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
49 readerBuffered = new BufferedReader(readerInputStream);
50
51 /* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_74b.java:49
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
46 return;
47 }
48 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
49 response.sendRedirect(data);
50 return;
51 }
52
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_74b.java, line 49 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_74a.
bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_74a.
java:52

```
49
50 /* read input from socket */
51
52 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
53 readerBuffered = new BufferedReader(readerInputStream);
54
55 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_74b.java, line 49 (Open Redirect) — Critical

Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_74b.java:49
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
46 return;
47 }
48 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
49 response.sendRedirect(data);
50 return;
51 }
52
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_21.java, line 56 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_21.bad_source
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_21.java:71

```
68 data = ""; /* initialize data in case id is not in query string */
69 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
70 {
71 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
72 while (tokenizer.hasMoreTokens())
73 {
74 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_21.java:56
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
53 return;
54 }
55 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
56 response.sendRedirect(data);
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_21.java, line 56 (Open Redirect) — Critical

```
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_07.java, line 66 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Servlet_07.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_07.java:41

```
if (privateFive == 5)
{
/* POTENTIAL FLAW: Read data from a querystring using getParameter */
data = request.getParameter("name");
}
else
{
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_07.java:66
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_41.java, line 54 (Open Redirect) — Critical

Issue Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_URLConnection_41.java, line 54 (Open Redirect) — Critical

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_41
.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_41
.java:74

```

try
{
readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_41.java:54
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getParameter_Servlet_61a.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_61b.badSource

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_61a.java, line 48 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_61b.java:31

```
String data;

/* POTENTIAL FLAW: Read data from a querystring using getParameter */
data = request.getParameter("name");

return data;
}
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_61a.java:48
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_08.java, line 138 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_08.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_08.java:66

```
connection = IO.getDBConnection();
/* prepare and execute a (hardcoded) query */
preparedStatement = connection.prepareStatement("select name from users where id=0");
resultSet = preparedStatement.executeQuery();
/* POTENTIAL FLAW: Read data from a database query resultset */
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_08.java, line 138 (Open Redirect) — Critical

```
data = resultSet.getString(1);
}
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_08.java:138
Taint Flags: DATABASE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_connect_tcp_31.java, line 130 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_31.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_31.j
ava:57

```

/* read input from socket */

readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_31.java:130
Taint Flags: NETWORK, NO_NEW_LINE, XSS

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_31.java, line 130 (Open Redirect) — Critical

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_66b.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl et_66a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl et_66a.java:34

```

/* Read data from cookies */
{
Cookie cookieSources[] = request.getCookies();
if (cookieSources != null)
{
/* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_66b.java:48
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_73b.java, line 49 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_73a.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_73a.j
ava:55

```
52
53  /* read input from socket */
54
55  readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
56  readerBuffered = new BufferedReader(readerInputStream);
57
58  /* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_73b.java:49
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
46  return;
47  }
48  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
49  response.sendRedirect(data);
50  return;
51  }
52
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_54e.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_54
a.bad

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_54e.java, line 47 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_54a.java:47

```
try
{
readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_54e.java:47
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_21.java, line 61 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_21.bad_source
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_21.java:84

```
/* Read data using an outbound tcp connection */
socket = new Socket("host.example.org", 39544);
/* read input from socket */
readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_connect_tcp_21.java, line 61 (Open Redirect) — Critical

```
86 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
87 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_21.java:61
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
58 return;
59 }
60 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
61 response.sendRedirect(data);
62 return;
63 }
64
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_02.java, line 124 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_02.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_02.java
:52

```
49 connection = IO.getDBConnection();
50 /* prepare and execute a (hardcoded) query */
51 preparedStatement = connection.prepareStatement("select name from users where id=0");
52 resultSet = preparedStatement.executeQuery();
53 /* POTENTIAL FLAW: Read data from a database query resultset */
54 data = resultSet.getString(1);
55 }
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_02.java:124
Taint Flags: DATABASE, XSS

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_02.java, line 124 (Open Redirect) — Critical

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_68b.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_68a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_68a.java:51

```

/* prepare and execute a (hardcoded) query */
preparedStatement = connection.prepareStatement("select name from users where id=0");
resultSet = preparedStatement.executeQuery();

/* POTENTIAL FLAW: Read data from a database query resultset */
data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_68b.java:47
Taint Flags: DATABASE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_51b.java, line 46 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_51a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_51a.jav
a:50

```
47
48  /* prepare and execute a (hardcoded) query */
49  preparedStatement = connection.prepareStatement("select name from users where id=0");
50  resultSet = preparedStatement.executeQuery();
51
52  /* POTENTIAL FLAW: Read data from a database query resultset */
53  data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_51b.java:46
Taint Flags: DATABASE, XSS

```
43  return;
44  }
45  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
46  response.sendRedirect(data);
47  return;
48  }
49
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_14.java, line 112 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_14
.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_14

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_14.java, line 112 (Open Redirect) — **Critical**

.java:49

```
InputStreamReader readerInputStream = null;
try
{
readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
/* POTENTIAL FLAW: Read data from a web server with URLConnection */
/* This will be reading the first "line" of the response body,
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_14.java:112
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_52c.java, line 47 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_52a.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_52a.j
ava:54

```

/* read input from socket */

readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_52c.java, line 47 (Open Redirect) — Critical

```
57 /* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_52c.java:47
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
44 return;
45 }
46 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47 response.sendRedirect(data);
48 return;
49 }
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_02.java, line 73 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_02.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_02.java:38

```
35 data = ""; /* initialize data in case id is not in query string */
36 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
37 {
38 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
39 while (tokenizer.hasMoreTokens())
40 {
41 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_02.java:73
Taint Flags: START_CHECKED_STRING, WEB, XSS

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_02.java, line 73 (Open Redirect) — Critical

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_68b.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_68a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_68a.java:51

```

/* read input from socket */

readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_68b.java:47
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_53d.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_Servlet_53a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_53a.java:35

```
32
33 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
34 {
35 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
36 while (tokenizer.hasMoreTokens())
37 {
38 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_53d.java:47
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
44 return;
45 }
46 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47 response.sendRedirect(data);
48 return;
49 }
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_17.java, line 68 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_17.java, line 68 (Open Redirect) — Critical

From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_17.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_17.java:38

```
35
36 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
37 {
38 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
39 while (tokenizer.hasMoreTokens())
40 {
41 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_17.java:68
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
65 return;
66 }
67 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
68 response.sendRedirect(data);
69 return;
70 }
71 }
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_15.java, line 75 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_15.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_15.java:40

```
37 data = ""; /* initialize data in case id is not in query string */
38 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_15.java, line 75 (Open Redirect) — Critical

```
{
StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
while (tokenizer.hasMoreTokens())
{
String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_15.java:75
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_07.java, line 130 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_07.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_07.java
:58

```
connection = IO.getDBConnection();
/* prepare and execute a (hardcoded) query */
preparedStatement = connection.prepareStatement("select name from users where id=0");
resultSet = preparedStatement.executeQuery();
/* POTENTIAL FLAW: Read data from a database query resultset */
data = resultSet.getString(1);
}
```

Sink Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_07.java, line 130 (Open Redirect) — Critical

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_07.java:130
Taint Flags: DATABASE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_listen_tcp_66b.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_66a.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_66a.j
ava:54

```

/* read input from socket */

readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_66b.java:48
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_listen_tcp_66b.java, line 48 (Open Redirect) — Critical

```
49 return;
50 }
51
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_01.java, line 137 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_01.ba
d
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_01.ja
va:57

```
54
55 /* read input from socket */
56
57 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
58 readerBuffered = new BufferedReader(readerInputStream);
59
60 /* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_01.java:137
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
134 return;
135 }
136 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
137 response.sendRedirect(data);
138 return;
139 }
140
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getQueryString_Servlet_81_bad.java, line 47 (Open Redirect) — Critical

Issue Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_81_bad.java, line 47 (Open Redirect) — **Critical**

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_Servlet_81a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_81a.java:35

```
32
33 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
34 {
35 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
36 while (tokenizer.hasMoreTokens())
37 {
38 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: action()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_81_bad.java:47
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
44 return;
45 }
46 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47 response.sendRedirect(data);
48 return;
49 }
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_45.java, line 58 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_45.bad

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_45.java, line 58 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_45.java:84

```
81
82 /* read input from socket */
83
84 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
85 readerBuffered = new BufferedReader(readerInputStream);
86
87 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_45.java:58
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
55 return;
56 }
57 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
58 response.sendRedirect(data);
59 return;
60 }
61
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_66b.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_Servlet_66a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_66a.java:35

```
32
33 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
34 {
35 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_66b.java, line 48 (Open Redirect) — **Critical**

```
while (tokenizer.hasMoreTokens())
{
String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_66b.java:48
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_67b.java, line 48 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_67a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_67a.java:59

```

/* read input from socket */

readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_67b.java, line 48 (Open Redirect) — Critical

Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_67b.java:48
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
45 return;
46 }
47 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48 response.sendRedirect(data);
49 return;
50 }
51
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_12.java, line 141 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_12.ba
d
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_12.ja
va:54

```
51 listener = new ServerSocket(39543);
52 socket = listener.accept();
53 /* read input from socket */
54 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
55 readerBuffered = new BufferedReader(readerInputStream);
56 /* POTENTIAL FLAW: Read data using a listening tcp connection */
57 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_12.java:141
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
138 return;
139 }
140 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
141 response.sendRedirect(data);
142 return;
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_12.java, line 141 (Open Redirect) — Critical

```
143 }
144
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_52c.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servlet_52a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_52a.java:34

```
31
32 /* Read data from cookies */
33 {
34 Cookie cookieSources[] = request.getCookies();
35 if (cookieSources != null)
36 {
37 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_52c.java:47
Taint Flags: WEB, XSS

```
44 return;
45 }
46 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47 response.sendRedirect(data);
48 return;
49 }
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_03.java, line 73 (Open Redirect) — Critical

Issue Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_03.java, line 73 (Open Redirect) — Critical

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_03.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_03.java:38

```
35 data = ""; /* initialize data in case id is not in query string */
36 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
37 {
38 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
39 while (tokenizer.hasMoreTokens())
40 {
41 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_03.java:73
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
70 return;
71 }
72 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
73 response.sendRedirect(data);
74 return;
75 }
76
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_45.java, line 52 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_45.bad

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getCookies_Servlet_45.java, line 52 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_45.java:67

```
64
65 /* Read data from cookies */
66 {
67 Cookie cookieSources[] = request.getCookies();
68 if (cookieSources != null)
69 {
70 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_45.java:52
Taint Flags: WEB, XSS

```
49 return;
50 }
51 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
52 response.sendRedirect(data);
53 return;
54 }
55
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_URLConnection_21.java, line 62 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_21
.bad_source
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_21
.java:82

```
79 InputStreamReader readerInputStream = null;
80 try
81 {
82 readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
83 readerBuffered = new BufferedReader(readerInputStream);
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_21.java, line 62 (Open Redirect) — Critical

```
84 /* POTENTIAL FLAW: Read data from a web server with URLConnection */
85 /* This will be reading the first "line" of the response body,
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_21.java:62
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
59 return;
60 }
61 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
62 response.sendRedirect(data);
63 return;
64 }
65
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_16.java, line 56 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_16.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_16.java:36

```
33 while (true)
34 {
35 /* POTENTIAL FLAW: Read data from a querystring using getParameter */
36 data = request.getParameter("name");
37 break;
38 }
39
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_16.java:56
Taint Flags: WEB, XSS

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_16.java, line 56 (Open Redirect) — Critical

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_81_bad.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Servlet_81a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_81a.java:31

```
String data;

/* POTENTIAL FLAW: Read data from a querystring using getParameter */
data = request.getParameter("name");

CWE601_Open_Redirect__Servlet_getParameter_Servlet_81_base baseObject = new CWE601_Open_Redirect__Servlet_getParameter_Servlet_81_bad();
baseObject.action(data , request, response);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: action()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_81_bad.java:47
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}
```

Open Redirect | **Critical**

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_81_bad.java, line 47 (Open Redirect) | **Critical**

```
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_12.java, line 74 (Open Redirect) | **Critical**

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_Servlet_12.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_12.java:38

```
35 data = ""; /* initialize data in case id is not in query string */
36 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
37 {
38 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
39 while (tokenizer.hasMoreTokens())
40 {
41 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_12.java:74
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
71 return;
72 }
73 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
74 response.sendRedirect(data);
75 return;
76 }
77
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_11.java, line 125 (Open Redirect) | **Critical**

Issue Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_11.java, line 125 (Open Redirect) — Critical

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_11.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_11.j
ava:51

```
48 /* Read data using an outbound tcp connection */
49 socket = new Socket("host.example.org", 39544);
50 /* read input from socket */
51 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
52 readerBuffered = new BufferedReader(readerInputStream);
53 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
54 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_11.java:125
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
122 return;
123 }
124 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
125 response.sendRedirect(data);
126 return;
127 }
128
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_81_bad.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_81a.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_81a.j

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_81_bad.java, line 47 (Open Redirect) — Critical

ava:54

```
51
52 /* read input from socket */
53
54 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
55 readerBuffered = new BufferedReader(readerInputStream);
56
57 /* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: action()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_81_bad.java:47
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
44 return;
45 }
46 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47 response.sendRedirect(data);
48 return;
49 }
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_45.java, line 59 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_45.ba
d
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_45.ja
va:87

```
84
85 /* read input from socket */
86
87 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
88 readerBuffered = new BufferedReader(readerInputStream);
89
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_45.java, line 59 (Open Redirect) — Critical

```
90 /* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_45.java:59
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
56 return;
57 }
58 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
59 response.sendRedirect(data);
60 return;
61 }
62
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_73b.java, line 49 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_73a.
bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_73a.
java:52

```
49
50 /* read input from socket */
51
52 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
53 readerBuffered = new BufferedReader(readerInputStream);
54
55 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_73b.java:49
Taint Flags: NETWORK, NO_NEW_LINE, XSS

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_73b.java, line 49 (Open Redirect) — Critical

```
46 return;
47 }
48 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
49 response.sendRedirect(data);
50 return;
51 }
52
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_10.java, line 68 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_10.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_10.java:37

```
34 data = ""; /* initialize data in case there are no cookies */
35 /* Read data from cookies */
36 {
37 Cookie cookieSources[] = request.getCookies();
38 if (cookieSources != null)
39 {
40 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_10.java:68
Taint Flags: WEB, XSS

```
65 return;
66 }
67 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
68 response.sendRedirect(data);
69 return;
70 }
71
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_21.java, line 55 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servlet_21.bad_source
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_21.java:70

```
67 data = ""; /* initialize data in case there are no cookies */
68 /* Read data from cookies */
69 {
70 Cookie cookieSources[] = request.getCookies();
71 if (cookieSources != null)
72 {
73 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_21.java:55
Taint Flags: WEB, XSS

```
52 return;
53 }
54 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
55 response.sendRedirect(data);
56 return;
57 }
58
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_06.java, line 131 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_06.bad

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_06.java, line 131 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_06.java:57

```
/* Read data using an outbound tcp connection */
socket = new Socket("host.example.org", 39544);
/* read input from socket */
readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
/* POTENTIAL FLAW: Read data using an outbound tcp connection */
data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_06.java:131
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_07.java, line 146 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_07.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_07.java:60

```
listener = new ServerSocket(39543);
socket = listener.accept();
/* read input from socket */
readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_07.java, line 146 (Open Redirect) — Critical

```
62 /* POTENTIAL FLAW: Read data using a listening tcp connection */
63 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_07.java:146
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
143 return;
144 }
145 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
146 response.sendRedirect(data);
147 return;
148 }
149
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_URLConnection_15.java, line 114 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_15
.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_15
.java:51

```
48 InputStreamReader readerInputStream = null;
49 try
50 {
51 readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
52 readerBuffered = new BufferedReader(readerInputStream);
53 /* POTENTIAL FLAW: Read data from a web server with URLConnection */
54 /* This will be reading the first "line" of the response body,
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_15.java:114
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_15.java, line 114 (Open Redirect) — Critical

```
111 return;
112 }
113 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
114 response.sendRedirect(data);
115 return;
116 }
117
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_45.java, line 58 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_45.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_45.java
:84

```
81
82 /* prepare and execute a (hardcoded) query */
83 preparedStatement = connection.prepareStatement("select name from users where id=0");
84 resultSet = preparedStatement.executeQuery();
85
86 /* POTENTIAL FLAW: Read data from a database query resultset */
87 data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_45.java:58
Taint Flags: DATABASE, XSS

```
55 return;
56 }
57 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
58 response.sendRedirect(data);
59 return;
60 }
61
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_03.java, line 124 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_03.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_03.java
:52

```
49  connection = IO.getDBConnection();
50  /* prepare and execute a (hardcoded) query */
51  preparedStatement = connection.prepareStatement("select name from users where id=0");
52  resultSet = preparedStatement.executeQuery();
53  /* POTENTIAL FLAW: Read data from a database query resultset */
54  data = resultSet.getString(1);
55  }
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_03.java:124
Taint Flags: DATABASE, XSS

```
121  return;
122  }
123  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
124  response.sendRedirect(data);
125  return;
126  }
127
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getParameter_Servlet_08.java, line 74 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_08.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_08.java, line 74 (Open Redirect) — Critical

vlet_08.java:49

```
if (privateReturnsTrue())
{
/* POTENTIAL FLAW: Read data from a querystring using getParameter */
data = request.getParameter("name");
}
else
{
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_08.java:74
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_31.java, line 115 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_31
.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_31
.java:53

```

try
{
readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_31.java, line 115 (Open Redirect) — Critical

```
56  /* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_31.java:115
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
112  return;
113  }
114  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
115  response.sendRedirect(data);
116  return;
117  }
118
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_02.java, line 112 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_02
.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_02
.java:49

```
46  InputStreamReader readerInputStream = null;
47  try
48  {
49  readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
50  readerBuffered = new BufferedReader(readerInputStream);
51  /* POTENTIAL FLAW: Read data from a web server with URLConnection */
52  /* This will be reading the first "line" of the response body,
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_02.java:112
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_02.java, line 112 (Open Redirect) — Critical

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_04.java, line 75 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_04.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_04.java:44

```
data = ""; /* initialize data in case there are no cookies */
/* Read data from cookies */
{
Cookie cookieSources[] = request.getCookies();
if (cookieSources != null)
{
/* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_04.java:75
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getCookies_Servlet_22a.java, line 55 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl et_22b.badSource
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl et_22b.java:35

```
32 data = ""; /* initialize data in case there are no cookies */
33 /* Read data from cookies */
34 {
35 Cookie cookieSources[] = request.getCookies();
36 if (cookieSources != null)
37 {
38 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_22a.java:55
Taint Flags: WEB, XSS

```
52 return;
53 }
54 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
55 response.sendRedirect(data);
56 return;
57 }
58
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getQueryString_Servlet_16.java, line 69 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S

Open Redirect | **Critical**

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_16.java, line 69 (Open Redirect) | **Critical**

ervlet_16.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_16.java:39

```
data = ""; /* initialize data in case id is not in query string */
/* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
{
StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
while (tokenizer.hasMoreTokens())
{
String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_16.java:69
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_71b.java, line 48 (Open Redirect) | **Critical**

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_71
a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_71
a.java:47

```

try
{
readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_71b.java, line 48 (Open Redirect) — Critical

```
48 readerBuffered = new BufferedReader(readerInputStream);
49
50 /* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_71b.java:48
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
45 return;
46 }
47 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48 response.sendRedirect(data);
49 return;
50 }
51
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_16.java, line 64 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_16.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_16.java:38

```
35 data = ""; /* initialize data in case there are no cookies */
36 /* Read data from cookies */
37 {
38 Cookie cookieSources[] = request.getCookies();
39 if (cookieSources != null)
40 {
41 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_16.java, line 64 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_16.java:64
Taint Flags: WEB, XSS

```
61 return;
62 }
63 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
64 response.sendRedirect(data);
65 return;
66 }
67
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_13.java, line 124 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_13.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_13.java
:52

```
49 connection = IO.getDBConnection();
50 /* prepare and execute a (hardcoded) query */
51 preparedStatement = connection.prepareStatement("select name from users where id=0");
52 resultSet = preparedStatement.executeQuery();
53 /* POTENTIAL FLAW: Read data from a database query resultset */
54 data = resultSet.getString(1);
55 }
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_13.java:124
Taint Flags: DATABASE, XSS

```
121 return;
122 }
123 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
124 response.sendRedirect(data);
125 return;
126 }
127
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_13.java, line 124 (Open Redirect) — **Critical**

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_01.java, line 120 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_01.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_01.java
:54

```
51
52  /* prepare and execute a (hardcoded) query */
53  preparedStatement = connection.prepareStatement("select name from users where id=0");
54  resultSet = preparedStatement.executeQuery();
55
56  /* POTENTIAL FLAW: Read data from a database query resultset */
57  data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_01.java:120
Taint Flags: DATABASE, XSS

```
117  return;
118  }
119  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
120  response.sendRedirect(data);
121  return;
122  }
123
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_16.java, line 121 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_16.java, line 121 (Open Redirect) — **Critical**

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_16.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_16.j
ava:52

```
49 /* Read data using an outbound tcp connection */
50 socket = new Socket("host.example.org", 39544);
51 /* read input from socket */
52 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
53 readerBuffered = new BufferedReader(readerInputStream);
54 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
55 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_16.java:121
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
118 return;
119 }
120 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
121 response.sendRedirect(data);
122 return;
123 }
124
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_08.java, line 139 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_08.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_08.j
ava:65

```
62 /* Read data using an outbound tcp connection */
63 socket = new Socket("host.example.org", 39544);
64 /* read input from socket */
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_08.java, line 139 (Open Redirect) — Critical

```
readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
/* POTENTIAL FLAW: Read data using an outbound tcp connection */
data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_08.java:139
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_61a.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_61b.bad
Source
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_61b.jav
a:51

```

/* prepare and execute a (hardcoded) query */
preparedStatement = connection.prepareStatement("select name from users where id=0");
resultSet = preparedStatement.executeQuery();

/* POTENTIAL FLAW: Read data from a database query resultset */
data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_61a.java, line 48 (Open Redirect) — Critical

Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_61a.java:48
Taint Flags: DATABASE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_41.java, line 54 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_41.ba
d
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_41.ja
va:81

```

/* read input from socket */

readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_41.java:54
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_41.java, line 54 (Open Redirect) — Critical

```
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_68b.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_68a.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_68a.j
ava:54

```

/* read input from socket */

readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_68b.java:47
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_61a.java, line 48 (Open Redirect) — Critical

Issue Details

Kingdom: Input Validation and Representation

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_61a.java, line 48 (Open Redirect) — Critical

Scan Engine: SCA (Data Flow)

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_61b.b
adSource
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_61b.j
ava:54

```
51
52  /* read input from socket */
53
54  readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
55  readerBuffered = new BufferedReader(readerInputStream);
56
57  /* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_61a.java:48
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
45  return;
46  }
47  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48  response.sendRedirect(data);
49  return;
50  }
51
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_51b.java, line 46 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_51a.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_51a.j
ava:53

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_51b.java, line 46 (Open Redirect) — Critical

```
/* read input from socket */

readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_51b.java:46
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_67b.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_67a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_67a.java:39

```

/* Read data from cookies */
{
Cookie cookieSources[] = request.getCookies();
if (cookieSources != null)
{
/* POTENTIAL FLAW: Read data from the first cookie value */
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_67b.java, line 48 (Open Redirect) — Critical

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_67b.java:48
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_11.java, line 73 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_Servlet_11.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_11.java:38

```
data = ""; /* initialize data in case id is not in query string */
/* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
{
StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
while (tokenizer.hasMoreTokens())
{
String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_11.java:73
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
return;
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_11.java, line 73 (Open Redirect) — Critical

```
71 }
72 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
73 response.sendRedirect(data);
74 return;
75 }
76
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_52c.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_52a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_52a.java:51

```
48
49 /* prepare and execute a (hardcoded) query */
50 preparedStatement = connection.prepareStatement("select name from users where id=0");
51 resultSet = preparedStatement.executeQuery();
52
53 /* POTENTIAL FLAW: Read data from a database query resultset */
54 data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_52c.java:47
Taint Flags: DATABASE, XSS

```
44 return;
45 }
46 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47 response.sendRedirect(data);
48 return;
49 }
50
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_42.java, line 127 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_42.badS
ource
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_42.java
:53

```
50
51  /* prepare and execute a (hardcoded) query */
52  preparedStatement = connection.prepareStatement("select name from users where id=0");
53  resultSet = preparedStatement.executeQuery();
54
55  /* POTENTIAL FLAW: Read data from a database query resultset */
56  data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_42.java:127
Taint Flags: DATABASE, XSS

```
124  return;
125  }
126  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
127  response.sendRedirect(data);
128  return;
129  }
130
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_11.java, line 68 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_11.bad

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getCookies_Servlet_11.java, line 68 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_11.java:37

```
34 data = ""; /* initialize data in case there are no cookies */
35 /* Read data from cookies */
36 {
37 Cookie cookieSources[] = request.getCookies();
38 if (cookieSources != null)
39 {
40 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_11.java:68
Taint Flags: WEB, XSS

```
65 return;
66 }
67 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
68 response.sendRedirect(data);
69 return;
70 }
71
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_database_74b.java, line 49 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_74a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_74a.java:52

```
49
50 /* prepare and execute a (hardcoded) query */
51 preparedStatement = connection.prepareStatement("select name from users where id=0");
52 resultSet = preparedStatement.executeQuery();
53
54 /* POTENTIAL FLAW: Read data from a database query resultset */
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

**testcases/CWE601_Open_Redirect/
CWE601_Open_Redirect__Servlet_database_74b.java, line 49 (Open Redirect)** — **Critical**

```
55  data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_74b.java:49
Taint Flags: DATABASE, XSS

```
46  return;
47  }
48  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
49  response.sendRedirect(data);
50  return;
51  }
52
```

**testcases/CWE601_Open_Redirect/
CWE601_Open_Redirect__Servlet_getQueryString_Servlet_41.java, line 48 (Open
Redirect)** — **Critical**

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_41.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_41.java:62

```
59
60  /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
61  {
62  StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
63  while (tokenizer.hasMoreTokens())
64  {
65  String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_41.java:48
Taint Flags: START_CHECKED_STRING, WEB, XSS

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_41.java, line 48 (Open Redirect) — **Critical**

```
45 return;
46 }
47 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48 response.sendRedirect(data);
49 return;
50 }
51
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_54e.java, line 47 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_54a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_54a.java:54

```
51
52 /* read input from socket */
53
54 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
55 readerBuffered = new BufferedReader(readerInputStream);
56
57 /* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_54e.java:47
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
44 return;
45 }
46 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47 response.sendRedirect(data);
48 return;
49 }
50
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_05.java, line 132 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_05.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_05.j
ava:58

```
55 /* Read data using an outbound tcp connection */
56 socket = new Socket("host.example.org", 39544);
57 /* read input from socket */
58 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
59 readerBuffered = new BufferedReader(readerInputStream);
60 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
61 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_05.java:132
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
129 return;
130 }
131 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
132 response.sendRedirect(data);
133 return;
134 }
135
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_68b.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_68
a.bad

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_68b.java, line 47 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_68
a.java:47

```
44
45  try
46  {
47  readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
48  readerBuffered = new BufferedReader(readerInputStream);
49
50  /* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_68b.java:47
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
44  return;
45  }
46  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47  response.sendRedirect(data);
48  return;
49  }
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_13.java, line 73 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_13.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_13.java:38

```
35  data = ""; /* initialize data in case id is not in query string */
36  /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
37  {
38  StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_13.java, line 73 (Open Redirect) — Critical

```
while (tokenizer.hasMoreTokens())
{
String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_13.java:73
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_68b.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_68a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_68a.java:34

```

/* Read data from cookies */
{
Cookie cookieSources[] = request.getCookies();
if (cookieSources != null)
{
/* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_68b.java, line 47 (Open Redirect) — Critical

Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_68b.java:47
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_10.java, line 124 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_10.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_10.java
:52

```
connection = IO.getDBConnection();
/* prepare and execute a (hardcoded) query */
preparedStatement = connection.prepareStatement("select name from users where id=0");
resultSet = preparedStatement.executeQuery();
/* POTENTIAL FLAW: Read data from a database query resultset */
data = resultSet.getString(1);
}
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_10.java:124
Taint Flags: DATABASE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_10.java, line 124 (Open Redirect) — Critical

```
127
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_41.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Servlet_41.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_41.java:58

```
55  String data;
56
57  /* POTENTIAL FLAW: Read data from a querystring using getParameter */
58  data = request.getParameter("name");
59
60  badSink(data , request, response );
61  }
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_41.java:47
Taint Flags: WEB, XSS

```
44  return;
45  }
46  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47  response.sendRedirect(data);
48  return;
49  }
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_01.java, line 122 (Open Redirect) — Critical

Issue Details

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

Open Redirect **Critical**

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_01.java, line 122 (Open Redirect) **Critical**

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_01.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_01.java:54

```
51
52 /* read input from socket */
53
54 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
55 readerBuffered = new BufferedReader(readerInputStream);
56
57 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_01.java:122
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
119 return;
120 }
121 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
122 response.sendRedirect(data);
123 return;
124 }
125
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_31.java, line 69 (Open Redirect) **Critical**

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servlet_31.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_31.java:40

```
37
38 /* Read data from cookies */
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_31.java, line 69 (Open Redirect) — **Critical**

```
39 {
40 Cookie cookieSources[] = request.getCookies();
41 if (cookieSources != null)
42 {
43 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_31.java:69
Taint Flags: WEB, XSS

```
66 return;
67 }
68 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
69 response.sendRedirect(data);
70 return;
71 }
72
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_02.java, line 68 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_02.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_02.java:37

```
34 data = ""; /* initialize data in case there are no cookies */
35 /* Read data from cookies */
36 {
37 Cookie cookieSources[] = request.getCookies();
38 if (cookieSources != null)
39 {
40 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getCookies_Servlet_02.java, line 68 (Open Redirect) — Critical

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_02.java:68
Taint Flags: WEB, XSS

```
65 return;
66 }
67 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
68 response.sendRedirect(data);
69 return;
70 }
71
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getQueryString_Servlet_73b.java, line 49 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S ervlet_73a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S ervlet_73a.java:36

```
33
34 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
35 {
36 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
37 while (tokenizer.hasMoreTokens())
38 {
39 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_73b.java:49
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
46 return;
47 }
48 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_73b.java, line 49 (Open Redirect) — Critical

```
49 response.sendRedirect(data);
50 return;
51 }
52
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_13.java, line 112 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_13
.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_13
.java:49

```
46 InputStreamReader readerInputStream = null;
47 try
48 {
49 readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
50 readerBuffered = new BufferedReader(readerInputStream);
51 /* POTENTIAL FLAW: Read data from a web server with URLConnection */
52 /* This will be reading the first "line" of the response body,
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_13.java:112
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
109 return;
110 }
111 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
112 response.sendRedirect(data);
113 return;
114 }
115
```

Open Redirect Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_67b.java, line 48 (Open Redirect) Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Servlet_67a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_67a.java:36

```
33 String data;
34
35 /* POTENTIAL FLAW: Read data from a querystring using getParameter */
36 data = request.getParameter("name");
37
38 Container dataContainer = new Container();
39 dataContainer.containerOne = data;
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_67b.java:48
Taint Flags: WEB, XSS

```
45 return;
46 }
47 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48 response.sendRedirect(data);
49 return;
50 }
51
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_52c.java, line 47 (Open Redirect) Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_52c.java, line 47 (Open Redirect) — Critical

From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_Servlet_52a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_52a.java:35

```
32
33 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
34 {
35 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
36 while (tokenizer.hasMoreTokens())
37 {
38 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_52c.java:47
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
44 return;
45 }
46 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47 response.sendRedirect(data);
48 return;
49 }
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_54e.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_Servlet_54a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_54a.java:35

```
32
33 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_54e.java, line 47 (Open Redirect) — Critical

```
{
StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
while (tokenizer.hasMoreTokens())
{
String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_54e.java:47
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_66b.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_66a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_66a.java:47

```

try
{
readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_66b.java, line 48 (Open Redirect) — Critical

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_66b.java:48
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
45 return;
46 }
47 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48 response.sendRedirect(data);
49 return;
50 }
51
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_03.java, line 125 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_03.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_03.j
ava:51

```
48 /* Read data using an outbound tcp connection */
49 socket = new Socket("host.example.org", 39544);
50 /* read input from socket */
51 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
52 readerBuffered = new BufferedReader(readerInputStream);
53 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
54 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_03.java:125
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
122 return;
123 }
124 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_03.java, line 125 (Open Redirect) — Critical

```
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_66b.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_66a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_66a.java:51

```

/* prepare and execute a (hardcoded) query */
preparedStatement = connection.prepareStatement("select name from users where id=0");
resultSet = preparedStatement.executeQuery();

/* POTENTIAL FLAW: Read data from a database query resultset */
data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_66b.java:48
Taint Flags: DATABASE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_81_bad.java, line 47 (Open Redirect) — Critical

Issue Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_81_bad.java, line 47 (Open Redirect) — Critical

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_81a.
bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_81a.
java:51

```
48
49 /* read input from socket */
50
51 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
52 readerBuffered = new BufferedReader(readerInputStream);
53
54 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: action()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_81_bad.java:47
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
44 return;
45 }
46 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47 response.sendRedirect(data);
48 return;
49 }
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_72b.java, line 49 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_72a.bad

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_72b.java, line 49 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_72a.java:32

```
String data;

/* POTENTIAL FLAW: Read data from a querystring using getParameter */
data = request.getParameter("name");

Vector<String> dataVector = new Vector<String>(5);
dataVector.add(0, data);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_72b.java:49
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_17.java, line 109 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_17.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_17.java:50

```

try
{
readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_17.java, line 109 (Open Redirect) — Critical

```
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_17.java:109
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}
}
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_05.java, line 75 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_05.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_05.java:44

```
data = ""; /* initialize data in case there are no cookies */
/* Read data from cookies */
{
Cookie cookieSources[] = request.getCookies();
if (cookieSources != null)
{
/* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_05.java, line 75 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_05.java:75
Taint Flags: WEB, XSS

```
72 return;
73 }
74 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
75 response.sendRedirect(data);
76 return;
77 }
78
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_42.java, line 144 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_42.ba
dSource
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_42.ja
va:56

```
53
54 /* read input from socket */
55
56 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
57 readerBuffered = new BufferedReader(readerInputStream);
58
59 /* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_42.java:144
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
141 return;
142 }
143 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
144 response.sendRedirect(data);
145 return;
146 }
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_42.java, line 144 (Open Redirect) — Critical

```
147
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_06.java, line 79 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_Servlet_06.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_06.java:44

```
41 data = ""; /* initialize data in case id is not in query string */
42 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
43 {
44 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
45 while (tokenizer.hasMoreTokens())
46 {
47 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_06.java:79
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
76 return;
77 }
78 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
79 response.sendRedirect(data);
80 return;
81 }
82
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_10.java, line 112 (Open Redirect) — Critical

Issue Details

Kingdom: Input Validation and Representation

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_10.java, line 112 (Open Redirect) — Critical

Scan Engine: SCA (Data Flow)

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_10
.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_10
.java:49

```
46  InputStreamReader readerInputStream = null;
47  try
48  {
49  readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
50  readerBuffered = new BufferedReader(readerInputStream);
51  /* POTENTIAL FLAW: Read data from a web server with URLConnection */
52  /* This will be reading the first "line" of the response body,
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_10.java:112
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
109  return;
110  }
111  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
112  response.sendRedirect(data);
113  return;
114  }
115
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_14.java, line 124 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_14.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_14.java
:52

Open Redirect | Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_14.java, line 124 (Open Redirect) | Critical

```
49 connection = IO.getDBConnection();
50 /* prepare and execute a (hardcoded) query */
51 preparedStatement = connection.prepareStatement("select name from users where id=0");
52 resultSet = preparedStatement.executeQuery();
53 /* POTENTIAL FLAW: Read data from a database query resultset */
54 data = resultSet.getString(1);
55 }
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_14.java:124
Taint Flags: DATABASE, XSS

```
121 return;
122 }
123 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
124 response.sendRedirect(data);
125 return;
126 }
127
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_61a.java, line 48 (Open Redirect) | Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_61b.badSource
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_61b.java:34

```
31
32 /* Read data from cookies */
33 {
34 Cookie cookieSources[] = request.getCookies();
35 if (cookieSources != null)
36 {
37 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getCookies_Servlet_61a.java, line 48 (Open Redirect) — Critical

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_61a.java:48
Taint Flags: WEB, XSS

```
45 return;
46 }
47 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48 response.sendRedirect(data);
49 return;
50 }
51
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getCookies_Servlet_06.java, line 74 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_06.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_06.java:43

```
40 data = ""; /* initialize data in case there are no cookies */
41 /* Read data from cookies */
42 {
43 Cookie cookieSources[] = request.getCookies();
44 if (cookieSources != null)
45 {
46 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_06.java:74
Taint Flags: WEB, XSS

```
71 return;
72 }
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_06.java, line 74 (Open Redirect) — **Critical**

```
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_68b.java, line 47 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Servlet_68a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_68a.java:31

```
{

/* POTENTIAL FLAW: Read data from a querystring using getParameter */
data = request.getParameter("name");

(new CWE601_Open_Redirect__Servlet_getParameter_Servlet_68b()).badSink(request, response);
}
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_68b.java:47
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

Open Redirect | Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_17.java, line 122 (Open Redirect) | Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_17.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_17.java
:54

```
51
52  /* prepare and execute a (hardcoded) query */
53  preparedStatement = connection.prepareStatement("select name from users where id=0");
54  resultSet = preparedStatement.executeQuery();
55
56  /* POTENTIAL FLAW: Read data from a database query resultset */
57  data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_17.java:122
Taint Flags: DATABASE, XSS

```
119  return;
120  }
121  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
122  response.sendRedirect(data);
123  return;
124  }
125  }
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_22a.java, line 55 (Open Redirect) | Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_22b.badSource

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_22a.java, line 55 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_22b.java:36

```
33 data = ""; /* initialize data in case id is not in query string */
34 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
35 {
36 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
37 while (tokenizer.hasMoreTokens())
38 {
39 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_22a.java:55
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
52 return;
53 }
54 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
55 response.sendRedirect(data);
56 return;
57 }
58
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_08.java, line 82 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servlet_08.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_08.java:51

```
48 data = ""; /* initialize data in case there are no cookies */
49 /* Read data from cookies */
50 {
51 Cookie cookieSources[] = request.getCookies();
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_08.java, line 82 (Open Redirect) — Critical

```
52 if (cookieSources != null)
53 {
54 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_08.java:82
Taint Flags: WEB, XSS

```
79 return;
80 }
81 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
82 response.sendRedirect(data);
83 return;
84 }
85
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_01.java, line 66 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_Servlet_01.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_01.java:38

```
35
36 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
37 {
38 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
39 while (tokenizer.hasMoreTokens())
40 {
41 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_01.java, line 66 (Open Redirect) — Critical

Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_01.java:66
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
63 return;
64 }
65 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
66 response.sendRedirect(data);
67 return;
68 }
69
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_53d.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_53
a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_53
a.java:47

```
44
45 try
46 {
47 readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
48 readerBuffered = new BufferedReader(readerInputStream);
49
50 /* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_53d.java:47
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
44 return;
45 }
46 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47 response.sendRedirect(data);
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_53d.java, line 47 (Open Redirect) — **Critical**

```
48 return;
49 }
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_72b.java, line 49 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_72a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_72a.java:48

```
45
46 try
47 {
48 readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
49 readerBuffered = new BufferedReader(readerInputStream);
50
51 /* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_72b.java:49
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
46 return;
47 }
48 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
49 response.sendRedirect(data);
50 return;
51 }
52
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_22a.java, line 55 (Open Redirect) — **Critical**

Issue Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_listen_tcp_22a.java, line 55 (Open Redirect) — Critical

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_22b.b
adSource
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_22b.j
ava:52

```
49 listener = new ServerSocket(39543);
50 socket = listener.accept();
51 /* read input from socket */
52 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
53 readerBuffered = new BufferedReader(readerInputStream);
54 /* POTENTIAL FLAW: Read data using a listening tcp connection */
55 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_22a.java:55
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
52 return;
53 }
54 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
55 response.sendRedirect(data);
56 return;
57 }
58
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getQueryString_Servlet_72b.java, line 49 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_72a.bad

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_72b.java, line 49 (Open Redirect) — **Critical**

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_72a.java:36

```
33
34 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
35 {
36 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
37 while (tokenizer.hasMoreTokens())
38 {
39 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_72b.java:49
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
46 return;
47 }
48 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
49 response.sendRedirect(data);
50 return;
51 }
52
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_31.java, line 145 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_31.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_31.java:60

```
57
58 /* read input from socket */
59
60 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_31.java, line 145 (Open Redirect) — **Critical**

```
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_31.java:145
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_52c.java, line 47 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_52a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_52a.java:51

```

/* read input from socket */

readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_52c.java, line 47 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_52c.java:47
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
44 return;
45 }
46 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47 response.sendRedirect(data);
48 return;
49 }
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_08.java, line 87 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_08.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_08.java:52

```
49 data = ""; /* initialize data in case id is not in query string */
50 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
51 {
52 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
53 while (tokenizer.hasMoreTokens())
54 {
55 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_08.java:87
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
84 return;
85 }
86 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
87 response.sendRedirect(data);
88 return;
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_08.java, line 87 (Open Redirect) — Critical

```
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_14.java, line 68 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl et_14.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl et_14.java:37

```
34 data = ""; /* initialize data in case there are no cookies */
35 /* Read data from cookies */
36 {
37 Cookie cookieSources[] = request.getCookies();
38 if (cookieSources != null)
39 {
40 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_14.java:68
Taint Flags: WEB, XSS

```
65 return;
66 }
67 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
68 response.sendRedirect(data);
69 return;
70 }
71
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_01.java, line 52 (Open Redirect) — Critical

Issue Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_01.java, line 52 (Open Redirect) — Critical

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser vlet_01.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser vlet_01.java:34

```
31  String data;
32
33  /* POTENTIAL FLAW: Read data from a querystring using getParameter */
34  data = request.getParameter("name");
35
36  if (data != null)
37  {
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_01.java:52
Taint Flags: WEB, XSS

```
49  return;
50  }
51  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
52  response.sendRedirect(data);
53  return;
54  }
55
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_51b.java, line 46 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S ervlet_51a.bad

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_51b.java, line 46 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_51a.java:34

```
/* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
{
StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
while (tokenizer.hasMoreTokens())
{
String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_51b.java:46
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_21.java, line 61 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_21.bad_source
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_21.java:85

```
connection = IO.getDBConnection();
/* prepare and execute a (hardcoded) query */
preparedStatement = connection.prepareStatement("select name from users where id=0");
resultSet = preparedStatement.executeQuery();
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_21.java, line 61 (Open Redirect) — Critical

```
/* POTENTIAL FLAW: Read data from a database query resultset */
data = resultSet.getString(1);
}
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_21.java:61
Taint Flags: DATABASE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_73b.java, line 49 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_73a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_73a.java:35

```

/* Read data from cookies */
{
Cookie cookieSources[] = request.getCookies();
if (cookieSources != null)
{
/* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getCookies_Servlet_73b.java, line 49 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_73b.java:49
Taint Flags: WEB, XSS

```
46 return;
47 }
48 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
49 response.sendRedirect(data);
50 return;
51 }
52
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getQueryString_Servlet_07.java, line 79 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S ervlet_07.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S ervlet_07.java:44

```
41 data = ""; /* initialize data in case id is not in query string */
42 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
43 {
44 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
45 while (tokenizer.hasMoreTokens())
46 {
47 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_07.java:79
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
76 return;
77 }
78 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
79 response.sendRedirect(data);
80 return;
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_07.java, line 79 (Open Redirect) — Critical

```
81 }
82
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_17.java, line 124 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_17.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_17.j
ava:54

```
51
52 /* read input from socket */
53
54 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
55 readerBuffered = new BufferedReader(readerInputStream);
56
57 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_17.java:124
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
121 return;
122 }
123 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
124 response.sendRedirect(data);
125 return;
126 }
127 }
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_31.java, line 74 (Open Redirect) — Critical

Issue Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_31.java, line 74 (Open Redirect) — Critical

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_Servlet_31.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_31.java:41

```
38
39  /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
40  {
41  StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
42  while (tokenizer.hasMoreTokens())
43  {
44  String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_31.java:74
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
71  return;
72  }
73  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
74  response.sendRedirect(data);
75  return;
76  }
77
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_22a.java, line 55 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_22b.badSource

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_22a.java, line 55 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_22b.java:47

```
InputStreamReader readerInputStream = null;
try
{
readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
/* POTENTIAL FLAW: Read data from a web server with URLConnection */
/* This will be reading the first "line" of the response body,
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_22a.java:55
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_54e.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servlet_54a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_54a.java:34

```

/* Read data from cookies */
{
Cookie cookieSources[] = request.getCookies();
if (cookieSources != null)
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_54e.java, line 47 (Open Redirect) — **Critical**

```
36 {
37 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_54e.java:47
Taint Flags: WEB, XSS

```
44 return;
45 }
46 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47 response.sendRedirect(data);
48 return;
49 }
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_04.java, line 80 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_Servlet_04.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_04.java:45

```
42 data = ""; /* initialize data in case id is not in query string */
43 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
44 {
45 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
46 while (tokenizer.hasMoreTokens())
47 {
48 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_04.java, line 80 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_04.java:80
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
77 return;
78 }
79 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
80 response.sendRedirect(data);
81 return;
82 }
83
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_74b.java, line 49 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_74a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_74a.java:36

```
33
34 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
35 {
36 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
37 while (tokenizer.hasMoreTokens())
38 {
39 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_74b.java:49
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
46 return;
47 }
48 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
49 response.sendRedirect(data);
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_74b.java, line 49 (Open Redirect) — Critical

```
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_21.java, line 55 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Servlet_21.bad_source
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_21.java:68

```
if (badPrivate)
{
/* POTENTIAL FLAW: Read data from a querystring using getParameter */
data = request.getParameter("name");
}
else
{
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_21.java:55
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_81_bad.java, line 47 (Open Redirect) — Critical

Issue Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_81_bad.java, line 47 (Open Redirect) — Critical

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_81a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_81a.jav
a:51

```
48
49  /* prepare and execute a (hardcoded) query */
50  preparedStatement = connection.prepareStatement("select name from users where id=0");
51  resultSet = preparedStatement.executeQuery();
52
53  /* POTENTIAL FLAW: Read data from a database query resultset */
54  data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: action()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_81_bad.java:47
Taint Flags: DATABASE, XSS

```
44  return;
45  }
46  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47  response.sendRedirect(data);
48  return;
49  }
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_09.java, line 125 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_09.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_09.j
ava:51

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_09.java, line 125 (Open Redirect) — **Critical**

```
48 /* Read data using an outbound tcp connection */
49 socket = new Socket("host.example.org", 39544);
50 /* read input from socket */
51 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
52 readerBuffered = new BufferedReader(readerInputStream);
53 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
54 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_09.java:125
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
122 return;
123 }
124 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
125 response.sendRedirect(data);
126 return;
127 }
128
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_72b.java, line 49 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_72a.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_72a.j
ava:55

```
52
53 /* read input from socket */
54
55 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
56 readerBuffered = new BufferedReader(readerInputStream);
57
58 /* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_72b.java, line 49 (Open Redirect) — **Critical**

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_72b.java:49
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
46 return;
47 }
48 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
49 response.sendRedirect(data);
50 return;
51 }
52
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_14.java, line 73 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_Servlet_14.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_14.java:38

```
35 data = ""; /* initialize data in case id is not in query string */
36 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
37 {
38 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
39 while (tokenizer.hasMoreTokens())
40 {
41 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_14.java:73
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
70 return;
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_14.java, line 73 (Open Redirect) — **Critical**

```
71 }
72 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
73 response.sendRedirect(data);
74 return;
75 }
76
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_12.java, line 126 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_12.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_12.j
ava:51

```
48 /* Read data using an outbound tcp connection */
49 socket = new Socket("host.example.org", 39544);
50 /* read input from socket */
51 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
52 readerBuffered = new BufferedReader(readerInputStream);
53 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
54 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_12.java:126
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
123 return;
124 }
125 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
126 response.sendRedirect(data);
127 return;
128 }
129
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_31.java, line 128 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_31.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_31.java
:57

```
54
55 /* prepare and execute a (hardcoded) query */
56 preparedStatement = connection.prepareStatement("select name from users where id=0");
57 resultSet = preparedStatement.executeQuery();
58
59 /* POTENTIAL FLAW: Read data from a database query resultset */
60 data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_31.java:128
Taint Flags: DATABASE, XSS

```
125 return;
126 }
127 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
128 response.sendRedirect(data);
129 return;
130 }
131
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_06.java, line 146 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_06.ba
d
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_06.ja

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_06.java, line 146 (Open Redirect) — Critical

va:60

```
listener = new ServerSocket(39543);
socket = listener.accept();
/* read input from socket */
readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
/* POTENTIAL FLAW: Read data using a listening tcp connection */
data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_06.java:146
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_05.java, line 131 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_05.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_05.java
:59

```
connection = IO.getDBConnection();
/* prepare and execute a (hardcoded) query */
preparedStatement = connection.prepareStatement("select name from users where id=0");
resultSet = preparedStatement.executeQuery();
/* POTENTIAL FLAW: Read data from a database query resultset */
data = resultSet.getString(1);
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_05.java, line 131 (Open Redirect) — **Critical**

```
62 }
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_05.java:131
Taint Flags: DATABASE, XSS

```
128 return;
129 }
130 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
131 response.sendRedirect(data);
132 return;
133 }
134
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_database_73b.java, line 49 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_73a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_73a.jav
a:52

```
49
50 /* prepare and execute a (hardcoded) query */
51 preparedStatement = connection.prepareStatement("select name from users where id=0");
52 resultSet = preparedStatement.executeQuery();
53
54 /* POTENTIAL FLAW: Read data from a database query resultset */
55 data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_73b.java:49
Taint Flags: DATABASE, XSS

```
46 return;
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_73b.java, line 49 (Open Redirect) — Critical

```
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_68b.java, line 74 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Servlet_68a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_68a.java:31

```
{

/* POTENTIAL FLAW: Read data from a querystring using getParameter */
data = request.getParameter("name");

(new CWE601_Open_Redirect__Servlet_getParameter_Servlet_68b()).badSink(request, response);
}
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: goodG2BSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_68b.java:74
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_15.java, line 70 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_15.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_15.java:39

```
36 data = ""; /* initialize data in case there are no cookies */
37 /* Read data from cookies */
38 {
39 Cookie cookieSources[] = request.getCookies();
40 if (cookieSources != null)
41 {
42 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_15.java:70
Taint Flags: WEB, XSS

```
67 return;
68 }
69 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
70 response.sendRedirect(data);
71 return;
72 }
73
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_53d.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_53d.java, line 47 (Open Redirect) — Critical

vlet_53a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_53a.java:31

```
String data;

/* POTENTIAL FLAW: Read data from a querystring using getParameter */
data = request.getParameter("name");

(new CWE601_Open_Redirect__Servlet_getParameter_Servlet_53b()).badSink(data , request, response);
}
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_53d.java:47
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_81_bad.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_81a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_81a.java:34

```

/* Read data from cookies */
{
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_81_bad.java, line 47 (Open Redirect) — Critical

```
Cookie cookieSources[] = request.getCookies();
if (cookieSources != null)
{
/* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: action()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_81_bad.java:47
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_11.java, line 140 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_11.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_11.java:54

```
listener = new ServerSocket(39543);
socket = listener.accept();
/* read input from socket */
readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
/* POTENTIAL FLAW: Read data using a listening tcp connection */
data = readerBuffered.readLine();
```

Sink Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_11.java, line 140 (Open Redirect) — Critical

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_11.java:140
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
137 return;
138 }
139 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
140 response.sendRedirect(data);
141 return;
142 }
143
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_67b.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_67a.
bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_67a.
java:56

```
53
54 /* read input from socket */
55
56 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
57 readerBuffered = new BufferedReader(readerInputStream);
58
59 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_67b.java:48
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
45 return;
46 }
47 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48 response.sendRedirect(data);
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_67b.java, line 48 (Open Redirect) — Critical

```
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_53d.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_53a.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_53a.j
ava:54

```

/* read input from socket */

readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_53d.java:47
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_42.java, line 59 (Open Redirect) — Critical

Issue Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getParameter_Servlet_42.java, line 59 (Open Redirect) — Critical

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser vlet_42.badSource
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser vlet_42.java:33

```
30 String data;
31
32 /* POTENTIAL FLAW: Read data from a querystring using getParameter */
33 data = request.getParameter("name");
34
35 return data;
36 }
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_42.java:59
Taint Flags: WEB, XSS

```
56 return;
57 }
58 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
59 response.sendRedirect(data);
60 return;
61 }
62
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_09.java, line 140 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_09.ba d
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_09.ja

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_09.java, line 140 (Open Redirect) — Critical

va:54

```
listener = new ServerSocket(39543);
socket = listener.accept();
/* read input from socket */
readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
/* POTENTIAL FLAW: Read data using a listening tcp connection */
data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_09.java:140
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_09.java, line 124 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_09.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_09.java
:52

```
connection = IO.getDBConnection();
/* prepare and execute a (hardcoded) query */
preparedStatement = connection.prepareStatement("select name from users where id=0");
resultSet = preparedStatement.executeQuery();
/* POTENTIAL FLAW: Read data from a database query resultset */
data = resultSet.getString(1);
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_09.java, line 124 (Open Redirect) — Critical

```
}
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_09.java:124
Taint Flags: DATABASE, XSS

```
121 return;
122 }
123 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
124 response.sendRedirect(data);
125 return;
126 }
127
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_42.java, line 73 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_42.badSource
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_42.java:37

```
34
35 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
36 {
37 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
38 while (tokenizer.hasMoreTokens())
39 {
40 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_42.java:73
Taint Flags: START_CHECKED_STRING, WEB, XSS

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_42.java, line 73 (Open Redirect) — Critical

```
70 return;
71 }
72 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
73 response.sendRedirect(data);
74 return;
75 }
76
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_71b.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl et_71a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl et_71a.java:34

```
31
32 /* Read data from cookies */
33 {
34 Cookie cookieSources[] = request.getCookies();
35 if (cookieSources != null)
36 {
37 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_71b.java:48
Taint Flags: WEB, XSS

```
45 return;
46 }
47 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48 response.sendRedirect(data);
49 return;
50 }
51
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_17.java, line 54 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Servlet_17.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_17.java:34

```
31  String data;
32
33  /* POTENTIAL FLAW: Read data from a querystring using getParameter */
34  data = request.getParameter("name");
35
36  for (int i = 0; i < 1; i++)
37  {
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_17.java:54
Taint Flags: WEB, XSS

```
51  return;
52  }
53  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
54  response.sendRedirect(data);
55  return;
56  }
57  }
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_61a.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_61a.java, line 48 (Open Redirect) — Critical

ervlet_61b.badSource
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_61b.java:35

```
32
33 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
34 {
35 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
36 while (tokenizer.hasMoreTokens())
37 {
38 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_61a.java:48
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
45 return;
46 }
47 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48 response.sendRedirect(data);
49 return;
50 }
51
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_71b.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_71a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_71a.java:35

```
32
33 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
34 {
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_71b.java, line 48 (Open Redirect) — Critical

```
35 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
36 while (tokenizer.hasMoreTokens())
37 {
38 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_71b.java:48
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
45 return;
46 }
47 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48 response.sendRedirect(data);
49 return;
50 }
51
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_68b.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_68a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_S
ervlet_68a.java:35

```
32
33 /* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
34 {
35 StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
36 while (tokenizer.hasMoreTokens())
37 {
38 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_68b.java, line 47 (Open Redirect) — Critical

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_68b.java:47
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
44 return;
45 }
46 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47 response.sendRedirect(data);
48 return;
49 }
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_05.java, line 147 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_05.ba
d
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_05.ja
va:61

```
58 listener = new ServerSocket(39543);
59 socket = listener.accept();
60 /* read input from socket */
61 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
62 readerBuffered = new BufferedReader(readerInputStream);
63 /* POTENTIAL FLAW: Read data using a listening tcp connection */
64 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_05.java:147
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
144 return;
145 }
146 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_05.java, line 147 (Open Redirect) — Critical

```
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_11.java, line 124 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_11.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_11.java
:52

```
49 connection = IO.getDBConnection();
50 /* prepare and execute a (hardcoded) query */
51 preparedStatement = connection.prepareStatement("select name from users where id=0");
52 resultSet = preparedStatement.executeQuery();
53 /* POTENTIAL FLAW: Read data from a database query resultset */
54 data = resultSet.getString(1);
55 }
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_11.java:124
Taint Flags: DATABASE, XSS

```
121 return;
122 }
123 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
124 response.sendRedirect(data);
125 return;
126 }
127
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_54e.java, line 47 (Open Redirect) — Critical

Issue Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_54e.java, line 47 (Open Redirect) — Critical

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_54a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_54a.java:31

```
28 String data;
29
30 /* POTENTIAL FLAW: Read data from a querystring using getParameter */
31 data = request.getParameter("name");
32
33 (new CWE601_Open_Redirect__Servlet_getParameter_Servlet_54b()).badSink(data , request, response);
34 }
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_54e.java:47
Taint Flags: WEB, XSS

```
44 return;
45 }
46 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47 response.sendRedirect(data);
48 return;
49 }
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_61a.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_61
b.badSource

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_61a.java, line 48 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_61
b.java:47

```
44
45 try
46 {
47 readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
48 readerBuffered = new BufferedReader(readerInputStream);
49
50 /* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_61a.java:48
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
45 return;
46 }
47 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48 response.sendRedirect(data);
49 return;
50 }
51
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_11.java, line 60 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_11.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_11.java:35

```
32 if (IO.staticReturnsTrue())
33 {
34 /* POTENTIAL FLAW: Read data from a querystring using getParameter */
35 data = request.getParameter("name");
36 }
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_11.java, line 60 (Open Redirect) — Critical

```
else
{
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_11.java:60
Taint Flags: WEB, XSS

```
57 return;
58 }
59 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
60 response.sendRedirect(data);
61 return;
62 }
63
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_09.java, line 68 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_09.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_09.java:37

```
34 data = ""; /* initialize data in case there are no cookies */
35 /* Read data from cookies */
36 {
37 Cookie cookieSources[] = request.getCookies();
38 if (cookieSources != null)
39 {
40 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_09.java:68
Taint Flags: WEB, XSS

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_09.java, line 68 (Open Redirect) — Critical

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_13.java, line 68 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servlet_13.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_13.java:37

```
data = ""; /* initialize data in case there are no cookies */
/* Read data from cookies */
{
Cookie cookieSources[] = request.getCookies();
if (cookieSources != null)
{
/* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_13.java:68
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_13.java, line 140 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_13.ba
d
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_13.ja
va:54

```
51 listener = new ServerSocket(39543);
52 socket = listener.accept();
53 /* read input from socket */
54 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
55 readerBuffered = new BufferedReader(readerInputStream);
56 /* POTENTIAL FLAW: Read data using a listening tcp connection */
57 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_13.java:140
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
137 return;
138 }
139 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
140 response.sendRedirect(data);
141 return;
142 }
143
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_02.java, line 140 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_02.ba
d

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_02.java, line 140 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_02.java:54

```
listener = new ServerSocket(39543);
socket = listener.accept();
/* read input from socket */
readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
/* POTENTIAL FLAW: Read data using a listening tcp connection */
data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_02.java:140
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_74b.java, line 49 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Servlet_74a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_74a.java:32

```
String data;

/* POTENTIAL FLAW: Read data from a querystring using getParameter */
data = request.getParameter("name");
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_74b.java, line 49 (Open Redirect) — Critical

```
33
34 HashMap<Integer,String> dataHashMap = new HashMap<Integer,String>();
35 dataHashMap.put(0, data);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_74b.java:49
Taint Flags: WEB, XSS

```
46 return;
47 }
48 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
49 response.sendRedirect(data);
50 return;
51 }
52
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_04.java, line 131 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_04.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_04.java
:59

```
56 connection = IO.getDBConnection();
57 /* prepare and execute a (hardcoded) query */
58 preparedStatement = connection.prepareStatement("select name from users where id=0");
59 resultSet = preparedStatement.executeQuery();
60 /* POTENTIAL FLAW: Read data from a database query resultset */
61 data = resultSet.getString(1);
62 }
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_04.java, line 131 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_04.java:131
Taint Flags: DATABASE, XSS

```
128 return;
129 }
130 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
131 response.sendRedirect(data);
132 return;
133 }
134
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_14.java, line 140 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_14.ba
d
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_14.ja
va:54

```
51 listener = new ServerSocket(39543);
52 socket = listener.accept();
53 /* read input from socket */
54 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
55 readerBuffered = new BufferedReader(readerInputStream);
56 /* POTENTIAL FLAW: Read data using a listening tcp connection */
57 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_14.java:140
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
137 return;
138 }
139 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
140 response.sendRedirect(data);
141 return;
142 }
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_14.java, line 140 (Open Redirect) — Critical

```
143
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_67b.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_67a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_67a.jav
a:56

```
53
54 /* prepare and execute a (hardcoded) query */
55 preparedStatement = connection.prepareStatement("select name from users where id=0");
56 resultSet = preparedStatement.executeQuery();
57
58 /* POTENTIAL FLAW: Read data from a database query resultset */
59 data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_67b.java:48
Taint Flags: DATABASE, XSS

```
45 return;
46 }
47 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48 response.sendRedirect(data);
49 return;
50 }
51
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_03.java, line 68 (Open Redirect) — Critical

Issue Details

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

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_03.java, line 68 (Open Redirect) — Critical

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl
et_03.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl
et_03.java:37

```
34 data = ""; /* initialize data in case there are no cookies */
35 /* Read data from cookies */
36 {
37 Cookie cookieSources[] = request.getCookies();
38 if (cookieSources != null)
39 {
40 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_03.java:68
Taint Flags: WEB, XSS

```
65 return;
66 }
67 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
68 response.sendRedirect(data);
69 return;
70 }
71
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_61a.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_61b.
badSource
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_61b.
java:51

```
48
49 /* read input from socket */
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_61a.java, line 48 (Open Redirect) — Critical

```

readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_61a.java:48
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_13.java, line 125 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_13.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_13.j
ava:51

```
/* Read data using an outbound tcp connection */
socket = new Socket("host.example.org", 39544);
/* read input from socket */
readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
/* POTENTIAL FLAW: Read data using an outbound tcp connection */
data = readerBuffered.readLine();
```

Sink Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_13.java, line 125 (Open Redirect) — Critical

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_13.java:125
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
122 return;
123 }
124 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
125 response.sendRedirect(data);
126 return;
127 }
128
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_12.java, line 113 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_12
.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_12
.java:49

```
46 InputStreamReader readerInputStream = null;
47 try
48 {
49 readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
50 readerBuffered = new BufferedReader(readerInputStream);
51 /* POTENTIAL FLAW: Read data from a web server with URLConnection */
52 /* This will be reading the first "line" of the response body,
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_12.java:113
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
110 return;
111 }
112 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
113 response.sendRedirect(data);
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_12.java, line 113 (Open Redirect) — Critical

```
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_71b.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_71a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_71a.java:51

```

/* read input from socket */

readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_71b.java:48
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_71b.java, line 48 (Open Redirect) — Critical

Issue Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_71b.java, line 48 (Open Redirect) — Critical

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Servlet_71a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_71a.java:31

```
28 String data;
29
30 /* POTENTIAL FLAW: Read data from a querystring using getParameter */
31 data = request.getParameter("name");
32
33 (new CWE601_Open_Redirect__Servlet_getParameter_Servlet_71b()).badSink((Object)data , request,
response );
34 }
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_71b.java:48
Taint Flags: WEB, XSS

```
45 return;
46 }
47 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48 response.sendRedirect(data);
49 return;
50 }
51
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_09.java, line 60 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getParameter_Servlet_09.java, line 60 (Open Redirect) — Critical

vlet_09.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_09.java:35

```
32  if (IO.STATIC_FINAL_TRUE)
33  {
34  /* POTENTIAL FLAW: Read data from a querystring using getParameter */
35  data = request.getParameter("name");
36  }
37  else
38  {
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_09.java:60
Taint Flags: WEB, XSS

```
57  return;
58  }
59  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
60  response.sendRedirect(data);
61  return;
62  }
63
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_04.java, line 147 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_04.ba
d
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_04.ja
va:61

```
58  listener = new ServerSocket(39543);
59  socket = listener.accept();
60  /* read input from socket */
61  readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_04.java, line 147 (Open Redirect) — Critical

```
readerBuffered = new BufferedReader(readerInputStream);
/* POTENTIAL FLAW: Read data using a listening tcp connection */
data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_04.java:147
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_10.java, line 125 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_10.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_10.j
ava:51

```
/* Read data using an outbound tcp connection */
socket = new Socket("host.example.org", 39544);
/* read input from socket */
readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);
/* POTENTIAL FLAW: Read data using an outbound tcp connection */
data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_10.java, line 125 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_10.java:125
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_52c.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Servlet_52a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_52a.java:31

```
String data;

/* POTENTIAL FLAW: Read data from a querystring using getParameter */
data = request.getParameter("name");

(new CWE601_Open_Redirect__Servlet_getParameter_Servlet_52b()).badSink(data , request, response);
}
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_52c.java:47
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getParameter_Servlet_52c.java, line 47 (Open Redirect) — **Critical**

```
49 }
50
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getParameter_Servlet_12.java, line 61 (Open Redirect) — **Critical**

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser vlet_12.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser vlet_12.java:35

```
32 if (IO.staticReturnsTrueOrFalse())
33 {
34 /* POTENTIAL FLAW: Read data from a querystring using getParameter */
35 data = request.getParameter("name");
36 }
37 else
38 {
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_12.java:61
Taint Flags: WEB, XSS

```
58 return;
59 }
60 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
61 response.sendRedirect(data);
62 return;
63 }
64
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_connect_tcp_53d.java, line 47 (Open Redirect) — **Critical**

Issue Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_53d.java, line 47 (Open Redirect) — Critical

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_53a.
bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_53a.
java:51

```
48
49 /* read input from socket */
50
51 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
52 readerBuffered = new BufferedReader(readerInputStream);
53
54 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_53d.java:47
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
44 return;
45 }
46 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47 response.sendRedirect(data);
48 return;
49 }
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_05.java, line 67 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_05.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser

Open Redirect	Critical
Package: testcases.CWE601_Open_Redirect	
testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getParameter_Servlet_05.java, line 67 (Open Redirect)	**Critical**

vlet_05.java:42

```
if (privateTrue)
{
/* POTENTIAL FLAW: Read data from a querystring using getParameter */
data = request.getParameter("name");
}
else
{
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_05.java:67
Taint Flags: WEB, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_listen_tcp_74b.java, line 49 (Open Redirect)	Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_74a.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_74a.j
ava:55

```

/* read input from socket */

readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_listen_tcp_74b.java, line 49 (Open Redirect) — Critical

```
58  /* POTENTIAL FLAW: Read data using a listening tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_74b.java:49
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
46  return;
47  }
48  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
49  response.sendRedirect(data);
50  return;
51  }
52
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getParameter_Servlet_15.java, line 62 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.ServletRequest.getParameter()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_15.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Ser
vlet_15.java:37

```
34  {
35  case 6:
36  /* POTENTIAL FLAW: Read data from a querystring using getParameter */
37  data = request.getParameter("name");
38  break;
39  default:
40  /* INCIDENTAL: CWE 561 Dead Code, the code below will never run
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_15.java:62
Taint Flags: WEB, XSS

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getParameter_Servlet_15.java, line 62 (Open Redirect) — Critical

```
59 return;
60 }
61 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
62 response.sendRedirect(data);
63 return;
64 }
65
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_07.java, line 131 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_07.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_07.java:57

```
54 /* Read data using an outbound tcp connection */
55 socket = new Socket("host.example.org", 39544);
56 /* read input from socket */
57 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
58 readerBuffered = new BufferedReader(readerInputStream);
59 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
60 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_07.java:131
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
128 return;
129 }
130 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
131 response.sendRedirect(data);
132 return;
133 }
134
```

Open Redirect

Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_getCookies_Servlet_51b.java, line 46 (Open Redirect)

Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servl et_51a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servl et_51a.java:33

```
30
31 /* Read data from cookies */
32 {
33 Cookie cookieSources[] = request.getCookies();
34 if (cookieSources != null)
35 {
36 /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_51b.java:46
Taint Flags: WEB, XSS

```
43 return;
44 }
45 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
46 response.sendRedirect(data);
47 return;
48 }
49
```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_database_53d.java, line 47 (Open Redirect)

Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_53a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_53a.jav

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_53d.java, line 47 (Open Redirect) — Critical

a:51

```

/* prepare and execute a (hardcoded) query */
preparedStatement = connection.prepareStatement("select name from users where id=0");
resultSet = preparedStatement.executeQuery();

/* POTENTIAL FLAW: Read data from a database query resultset */
data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_53d.java:47
Taint Flags: DATABASE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_67b.java, line 48 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getQueryString()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getQueryString_Servlet_67a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_67a.java:40

```

/* POTENTIAL FLAW: Parse id param out of the URL querystring (without using getParameter()) */
{
StringTokenizer tokenizer = new StringTokenizer(request.getQueryString(), "&");
while (tokenizer.hasMoreTokens())
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_67b.java, line 48 (Open Redirect) — Critical

```
42 {
43 String token = tokenizer.nextToken(); /* a token will be like "id=foo" */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getQueryString_Servlet_67b.java:48
Taint Flags: START_CHECKED_STRING, WEB, XSS

```
45 return;
46 }
47 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
48 response.sendRedirect(data);
49 return;
50 }
51
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_54e.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_54a.
bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_54a.
java:51

```
48
49 /* read input from socket */
50
51 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
52 readerBuffered = new BufferedReader(readerInputStream);
53
54 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_54e.java, line 47 (Open Redirect) — Critical

File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_54e.java:47
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_54e.java, line 47 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.sql.PreparedStatement.executeQuery()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_database_54a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_54a.jav
a:51

```

/* prepare and execute a (hardcoded) query */
preparedStatement = connection.prepareStatement("select name from users where id=0");
resultSet = preparedStatement.executeQuery();

/* POTENTIAL FLAW: Read data from a database query resultset */
data = resultSet.getString(1);
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_54e.java:47
Taint Flags: DATABASE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_database_54e.java, line 47 (Open Redirect) — Critical

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_02.java, line 125 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_02.b
ad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_02.j
ava:51

```
48 /* Read data using an outbound tcp connection */
49 socket = new Socket("host.example.org", 39544);
50 /* read input from socket */
51 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
52 readerBuffered = new BufferedReader(readerInputStream);
53 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
54 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_02.java:125
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
122 return;
123 }
124 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
125 response.sendRedirect(data);
126 return;
127 }
128
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_52c.java, line 47 (Open Redirect) — Critical

Issue Details

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

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_52c.java, line 47 (Open Redirect) — Critical

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_52
a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_52
a.java:47

```
44
45 try
46 {
47 readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
48 readerBuffered = new BufferedReader(readerInputStream);
49
50 /* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_52c.java:47
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
44 return;
45 }
46 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47 response.sendRedirect(data);
48 return;
49 }
50
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_21.java, line 62 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_listen_tcp_21.ba
d_source
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_21.ja
va:87

```
84 listener = new ServerSocket(39543);
85 socket = listener.accept();
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_21.java, line 62 (Open Redirect) — Critical

```
86 /* read input from socket */
87 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
88 readerBuffered = new BufferedReader(readerInputStream);
89 /* POTENTIAL FLAW: Read data using a listening tcp connection */
90 data = readerBuffered.readLine();
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_listen_tcp_21.java:62
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
59 return;
60 }
61 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
62 response.sendRedirect(data);
63 return;
64 }
65
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_72b.java, line 49 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.Socket.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_connect_tcp_72a.
bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_72a.
java:52

```
49
50 /* read input from socket */
51
52 readerInputStream = new InputStreamReader(socket.getInputStream(), "UTF-8");
53 readerBuffered = new BufferedReader(readerInputStream);
54
55 /* POTENTIAL FLAW: Read data using an outbound tcp connection */
```

Sink Details

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_connect_tcp_72b.java, line 49 (Open Redirect) — Critical

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_connect_tcp_72b.java:49
Taint Flags: NETWORK, NO_NEW_LINE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
return;
}

```

testcases/CWE601_Open_Redirect/ CWE601_Open_Redirect__Servlet_URLConnection_42.java, line 114 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_42
.badSource
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_42
.java:49

```

try
{
readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
readerBuffered = new BufferedReader(readerInputStream);

/* POTENTIAL FLAW: Read data from a web server with URLConnection */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_42.java:114
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
return;
}
/* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
response.sendRedirect(data);
```

Open Redirect — Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_42.java, line 114 (Open Redirect) — Critical

```
return;
}

```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_08.java, line 126 (Open Redirect) — Critical

Issue Details

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

Source Details

Source: java.net.URLConnection.getInputStream()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_URLConnection_08
.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_08
.java:63

```
60 InputStreamReader readerInputStream = null;
61 try
62 {
63 readerInputStream = new InputStreamReader(urlConnection.getInputStream(), "UTF-8");
64 readerBuffered = new BufferedReader(readerInputStream);
65 /* POTENTIAL FLAW: Read data from a web server with URLConnection */
66 /* This will be reading the first "line" of the response body,
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: bad()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_URLConnection_08.java:126
Taint Flags: NO_NEW_LINE, WEBSERVICE, XSS

```
123 return;
124 }
125 /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
126 response.sendRedirect(data);
127 return;
128 }
129
```

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_53d.java, line 47 (Open Redirect) — Critical

Issue Details

Open Redirect | Critical

Package: testcases.CWE601_Open_Redirect

testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_53d.java, line 47 (Open Redirect) | Critical

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

Source Details

Source: javax.servlet.http.HttpServletRequest.getCookies()
From: testcases.CWE601_Open_Redirect.CWE601_Open_Redirect__Servlet_getCookies_Servlet_53a.bad
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_53a.java:34

```
31
32  /* Read data from cookies */
33  {
34  Cookie cookieSources[] = request.getCookies();
35  if (cookieSources != null)
36  {
37  /* POTENTIAL FLAW: Read data from the first cookie value */
```

Sink Details

Sink: javax.servlet.http.HttpServletResponse.sendRedirect()
Enclosing Method: badSink()
File: testcases/CWE601_Open_Redirect/CWE601_Open_Redirect__Servlet_getCookies_Servlet_53d.java:47
Taint Flags: WEB, XSS

```
44  return;
45  }
46  /* POTENTIAL FLAW: redirect is sent verbatim; escape the string to prevent ancillary issues like XSS, Response splitting etc */
47  response.sendRedirect(data);
48  return;
49  }
50
```