

Security Report

Scan Name: Demo SAST

Technology: SAST

Report Name: Demo-SAST

Report created at: Monday, March 22, 2021

Notes: Sample report for demo application.

Summary of security issues

High severity issues:	72
Medium severity issues:	8
Low severity issues:	1
<hr/>	
Total security issues:	81

Scan Information

Scan started: Wednesday, January 8, 2020 5:06:25 AM (UTC)

Table of Contents

Summary

- [Issues](#)

Fix-Groups

- [Common API Call: Authentication.Entity](#): java.sql.DriverManager.getConnection(java.lang.String):java.sql.Connection
- [Common API Call: CrossSiteScripting](#): javax.servlet.jsp.JspWriter.print(java.lang.Object):void
- [Common API Call: CrossSiteScripting](#): javax.servlet.jsp.JspWriter.print(java.lang.String):void

- Common API Call: `CrossSiteScripting.Reflected`: Insecure Use of `Document.Location`
- Common API Call: `CrossSiteScripting.Reflected`: Insecure Use of `Document.Write`
- Common API Call: `CrossSiteScripting.Reflected`: Insecure Use of `InnerHTML` or `OuterHTML`
- Common API Call: `CrossSiteScripting.Reflected`: Insecure Use of `setAttribute`
- Common API Call: `CrossSiteScripting.Reflected`: Potential Issue With Included Script
- Common API Call: `Validation.EncodingRequired`: `java.io.PrintWriter.write(java.lang.String):void`
- Common API Call: `Validation.Required`: `javax.servlet.http.HttpSession.setAttribute(java.lang.String;java.lang.Object):void`
- Common API Call: `Validation.Required`: `javax.servlet.ServletRequest.setAttribute(java.lang.String;java.lang.Object):void`
- Common Open Source: `OpenSource`: `commons-codec-1.6.jar`
- Common Open Source: `OpenSource`: `derby.jar`
- Common Open Source: `OpenSource`: `swfobject.js`
- Common Fix Point: `Injection.SQL`:
`com.ibm.security.appscan.altoromutual.model.User.getUserTransactions(java.lang.String;java.lang.String;com.ibm.security.appscan.altoromutual.model.Account[]):com.ibm.security.appscan.altoromutual.model.Transaction[]`
- Common Fix Point: `Injection.SQL`:
`com.ibm.security.appscan.altoromutual.util.DBUtil.addAccount(java.lang.String;java.lang.String):java.lang.String`
- Common Fix Point: `Injection.SQL`:
`com.ibm.security.appscan.altoromutual.util.DBUtil.addUser(java.lang.String;java.lang.String;java.lang.String;java.lang.String):java.lang.String`
- Common Fix Point: `Injection.SQL`:
`com.ibm.security.appscan.altoromutual.util.DBUtil.changePassword(java.lang.String;java.lang.String):java.lang.String`
- Common Fix Point: `Injection.SQL`: `com.ibm.security.appscan.altoromutual.util.DBUtil.isValidUser(java.lang.String;java.lang.String):boolean`
- Common Fix Point: `CrossSiteScripting`: `java.lang.StringBuilder.toString():java.lang.String`
- Common Fix Point: `CrossSiteScripting`: `java.util.ArrayList.toArray(java.lang.Object[]):java.lang.Object[]`
- Common Fix Point: `Injection.SQL`: `java.lang.StringBuilder.append(java.lang.String):java.lang.StringBuilder`
- Common Fix Point: `Validation.Required`: `javax.servlet.http.HttpSession.setAttribute(java.lang.String;java.lang.Object):void`
- Common Fix Point: `Validation.Required.URL.Redirect`: `javax.servlet.http.HttpServletRequest.getContextPath():java.lang.String`

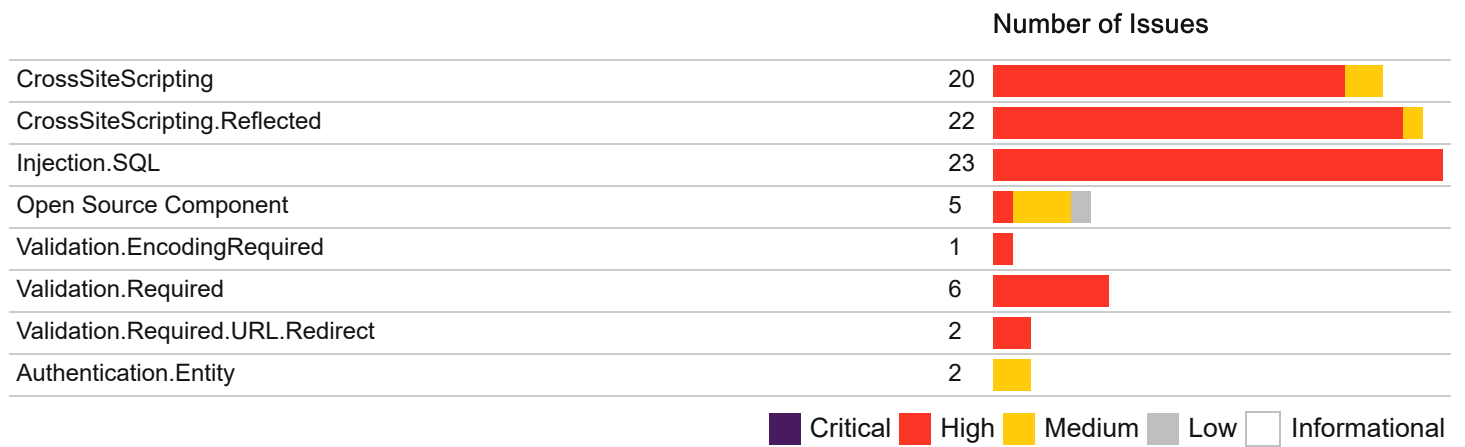
How to Fix

- Cross-site Scripting
- Reflected Cross-site Scripting
- Vulnerable Open Source Component
- SQL Injection
- `Validation.EncodingRequired`
- `Validation.Required`
- Unvalidated Redirect
- `Authentication.Entity`
- Reflected Cross-site Scripting
- Insecure Use of `Document.Location`
- Reflected Cross-site Scripting
- Reflected Cross-site Scripting
- Reflected Cross-site Scripting
- `Authentication.Entity`

Summary

Total security issues: 81

Issue Types: 8



Issues - By Fix Groups:

M	Common API Call: Authentication.Entity: java.sql.DriverManager.getConnection(java.lang.String):java.sql...
Fix Group ID:	b5d99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
API:	java.sql.DriverManager.getConnection(java.lang.String):java.sql.Connection
Notes:	
How to Fix:	Authentication.Entity

Issue 1 of 2

Issue ID:	29da9bc8-ca7f-eb11-85aa-281878e650eb
Severity:	Medium
Status	Noise
Classification	Definitive
Fix Group ID:	b5d99bc8-ca7f-eb11-85aa-281878e650eb
Location	com.ibm.security.appscan.altoromutual.util.DBUtil.getConnection():Connection
Line	83
Source File	com\ibm\security\appscan\altoromutual\util\DBUtil.class
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	255
API:	java.sql.DriverManager.getConnection(String):Connection
Caller:	com.ibm.security.appscan.altoromutual.util.DBUtil.getConnection():Connection at line 83

Issue 1 of 2 - Details

Call

```
getConnection("jdbc:derby:altoro")
```

Issue 2 of 2

Issue ID:	2cda9bc8-ca7f-eb11-85aa-281878e650eb
Severity:	Medium
Status	Noise
Classification	Definitive
Fix Group ID:	b5d99bc8-ca7f-eb11-85aa-281878e650eb
Location	com.ibm.security.appscan.altoromutual.util.DBUtil.initDB():void
Line	102
Source File	com\ibm\security\appscan\altoromutual\util\DBUtil.class
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	255
API:	java.sql.DriverManager.getConnection(String):Connection
Caller:	com.ibm.security.appscan.altoromutual.util.DBUtil.initDB():void at line 102

Issue 2 of 2 - Details

Call

```
getConnection("jdbc:derby:altoro;create=true")
```

M Common API Call: CrossSiteScripting:
javax.servlet.jsp.JspWriter.print(java.lang.Object):void

Fix Group ID:	b6d99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
API:	javax.servlet.jsp.JspWriter.print(java.lang.Object):void
Notes:	
How to Fix:	Cross-site Scripting

Issue 1 of 2

Issue ID:	b61794ce-ca7f-eb11-85aa-281878e650eb
Severity:	Medium
Status	Noise
Classification	Definitive
Fix Group ID:	b6d99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(Object):void subscribe_jsp:40
Line	40
Source File	subscribe_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.ServletRequest.getAttribute(String):Object via subscribe_jsp:40
Sink:	javax.servlet.jsp.JspWriter.print(Object):void via subscribe_jsp:40

Issue 1 of 2 - Details

Trace

```

org.apache.jsp.subscribe_jsp._jspService(HttpServletRequest;HttpServletResponse):void
┆
┆ org.apache.jsp.subscribe_jsp:40
┆ request.getAttribute("message_subscribe")
┆
┆ org.apache.jsp.subscribe_jsp:40
┆ Temp#9@0.print(Temp#10@0)




```

Issue 2 of 2

Issue ID:	a11794ce-ca7f-eb11-85aa-281878e650eb
Severity:	Medium
Status	Noise
Classification	Definitive
Fix Group ID:	b6d99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(Object):void transfer_jsp:101
Line	101
Source File	transfer_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.ServletRequest.getAttribute(String):Object via transfer_jsp:101
Sink:	javax.servlet.jsp.JspWriter.print(Object):void via transfer_jsp:101

Issue 2 of 2 - Details

Trace

	org.apache.jsp.bank.transfer_jsp._jspService(HttpServletRequest;HttpServletResponse):void
	org.apache.jsp.bank.transfer_jsp:101 request.getAttribute("message")
	org.apache.jsp.bank.transfer_jsp:101 Temp#17@0.print(Temp#18@0)






H	Common API Call: CrossSiteScripting: javax.servlet.jsp.JspWriter.print(java.lang.String):void
Fix Group ID:	add99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
API:	javax.servlet.jsp.JspWriter.print(java.lang.String):void
Notes:	
How to Fix:	Cross-site Scripting

Issue 1 of 14

Issue ID:	891794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	add99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void login_jsp:47
Line	47
Source File	login_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.ServletRequest.getAttribute(String):Object <i>via</i> login_jsp:43
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> login_jsp:47

Issue 1 of 14 - Details

Trace




	org.apache.jsp.admin.login_jsp._jspService(HttpServletRequest;HttpServletResponse):void
	
	org.apache.jsp.admin.login_jsp:43 error = request. getAttribute ("loginError")
	
	org.apache.jsp.admin.login_jsp:47 out.print(error)

Issue 2 of 14

Issue ID:	801794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	add99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void admin_jsp:61
Line	61
Source File	admin_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.http.HttpSession.getAttribute(String):Object <i>via</i> admin_jsp:58
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> admin_jsp:61

Issue 2 of 14 - Details

Trace




	org.apache.jsp.admin.admin_jsp._jspService(HttpServletRequest;HttpServletResponse):void
	org.apache.jsp.admin.admin_jsp:58 error = request.getSession(). getAttribute ("message")
	org.apache.jsp.admin.admin_jsp:61 out.print(error)

Issue 3 of 14

Issue ID:	8c1794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	add99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void balance_jsp:57
Line	57
Source File	balance_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> balance_jsp:40
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> balance_jsp:57

Issue 3 of 14 - Details

Trace


	org.apache.jsp.bank.balance_jsp._jspService(HttpServletRequest;HttpServletResponse):void
	org.apache.jsp.bank.balance_jsp:40 paramName = request. getParameter ("acctId")
	org.apache.jsp.bank.balance_jsp:57 out.print(accountName)

Issue 4 of 14

Issue ID:	8f1794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	add99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void customize_jsp:35
Line	35
Source File	customize_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> customize_jsp:35
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> customize_jsp:35

Issue 4 of 14 - Details

Trace




	org.apache.jsp.bank.customize_jsp._jspService(HttpServletRequest;HttpServletResponse):void
	org.apache.jsp.bank.customize_jsp:35 request. getParameter ("lang")
	org.apache.jsp.bank.customize_jsp:35 Temp#9@0.print(Temp#10@0)

Issue 5 of 14

Issue ID:	9b1794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	add99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void transaction_jsp:115
Line	115
Source File	transaction_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> transaction_jsp:115
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> transaction_jsp:115

Issue 5 of 14 - Details

Trace




	org.apache.jsp.bank.transaction_jsp._jspService(HttpServletRequest;HttpServletResponse):void
	org.apache.jsp.bank.transaction_jsp:115 request. getParameter ("startDate")
	org.apache.jsp.bank.transaction_jsp:115 Temp#19@0.print(Temp#20@0)

Issue 6 of 14

Issue ID:	9e1794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	add99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void transaction_jsp:117
Line	117
Source File	transaction_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> transaction_jsp:117
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> transaction_jsp:117

Issue 6 of 14 - Details

Trace




	org.apache.jsp.bank.transaction_jsp._jspService(HttpServletRequest;HttpServletResponse):void
	org.apache.jsp.bank.transaction_jsp:117 request. getParameter ("endDate")
	org.apache.jsp.bank.transaction_jsp:117 Temp#21@0.print(Temp#22@0)

Issue 7 of 14

Issue ID:	b31794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	add99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void search_jsp:44
Line	44
Source File	search_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> search_jsp:32
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> search_jsp:44

Issue 7 of 14 - Details

Trace




	org.apache.jsp.search_jsp._jspService(HttpServletRequest;HttpServletResponse):void
	org.apache.jsp.search_jsp:32 query = request. getParameter ("query")
	org.apache.jsp.search_jsp:44 out.print(query)

Issue 8 of 14

Issue ID:	981794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	add99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void queryxpath_jsp:36
Line	36
Source File	queryxpath_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> queryxpath_jsp:36
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> queryxpath_jsp:36

Issue 8 of 14 - Details

Trace




	org.apache.jsp.bank.queryxpath_jsp._jspService(HttpServletRequest;HttpServletResponse):void
	org.apache.jsp.bank.queryxpath_jsp:36 request. getParameter ("query")
	org.apache.jsp.bank.queryxpath_jsp:36 Temp#13@0.print(Temp#14@0)

Issue 9 of 14

Issue ID:	b91794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	add99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void serverStatusCheckService_jsp:4
Line	4
Source File	serverStatusCheckService_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> serverStatusCheckService_jsp:4
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> serverStatusCheckService_jsp:4

Issue 9 of 14 - Details

Trace

	org.apache.jsp.util.serverStatusCheckService_jsp._jspService(HttpServletRequest;HttpServletResponse):void
	org.apache.jsp.util.serverStatusCheckService_jsp:4 request. getParameter ("HostName")
	org.apache.jsp.util.serverStatusCheckService_jsp:4 out.print(request.getParameter ("HostName"))

Issue 10 of 14

Issue ID:	a41794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	add99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void feedback_jsp:59
Line	59
Source File	feedback_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.http.HttpSession.getAttribute(String):Object <i>via</i> feedback_jsp:33
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> feedback_jsp:59

Issue 10 of 14 - Details

Trace

<input type="checkbox"/>	org.apache.jsp.feedback_jsp._jspService(HttpServletRequest;HttpServletResponse):void
<input checked="" type="checkbox"/>	org.apache.jsp.feedback_jsp:33 user = request.getSession(). getAttribute ("user")
<input type="checkbox"/>	org.apache.jsp.feedback_jsp:59 user .getLastName()
<input type="checkbox"/>	org.apache.jsp.feedback_jsp:59 Temp#19@0 = Temp#19@0.append(Temp#20@0)
<input type="checkbox"/>	org.apache.jsp.feedback_jsp:59 Temp#19@0.append (Temp#20@0).toString()
<input checked="" type="checkbox"/>	org.apache.jsp.feedback_jsp:59 Temp#18@0.print(Temp#19@0.append (Temp#20@0).toString())

Issue 11 of 14

Issue ID:	a71794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	add99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void feedbacksuccess_jsp:40
Line	40
Source File	feedbacksuccess_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.ServletRequest.getAttribute(String):Object <i>via</i> feedbacksuccess_jsp:40
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> feedbacksuccess_jsp:40

Issue 11 of 14 - Details

Trace

```

org.apache.jsp.feedbacksuccess_jsp._jspService(HttpServletRequest;HttpServletResponse):void
├── org.apache.jsp.feedbacksuccess_jsp:40
│   │ request.getAttribute("message_feedback")
│   └── org.apache.jsp.feedbacksuccess_jsp:40
│       │ new StringBuilder().append(request.getAttribute("message_feedback"))
│       └── org.apache.jsp.feedbacksuccess_jsp:40
│           │ new StringBuilder().append(request.getAttribute("message_feedback")).toString()
│           └── org.apache.jsp.feedbacksuccess_jsp:40
│               │ Temp#10@0.print(Temp#11@0)
└──

```

Issue 12 of 14

Issue ID:	aa1794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	add99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void feedbacksuccess_jsp:44
Line	44
Source File	feedbacksuccess_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> feedbacksuccess_jsp:41
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> feedbacksuccess_jsp:44

Issue 12 of 14 - Details

Trace

<input type="checkbox"/>	org.apache.jsp.feedbacksuccess_jsp._jspService(HttpServletRequest;HttpServletResponse):void
<input checked="" type="checkbox"/>	org.apache.jsp.feedbacksuccess_jsp:41 email = request. getParameter ("email_addr")
<input checked="" type="checkbox"/>	org.apache.jsp.feedbacksuccess_jsp:44 email .toLowerCase()
<input type="checkbox"/>	org.apache.jsp.feedbacksuccess_jsp:44 sanitzieHtmlWithRegex(email.toLowerCase())
<input checked="" type="checkbox"/>	org.apache.jsp.feedbacksuccess_jsp:44 out.print(sanitzieHtmlWithRegex(email.toLowerCase()))

Issue 13 of 14

Issue ID:	ad1794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	add99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void feedbacksuccess_jsp:46
Line	46
Source File	feedbacksuccess_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> feedbacksuccess_jsp:41
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> feedbacksuccess_jsp:46

Issue 13 of 14 - Details

Trace

<input type="checkbox"/>	org.apache.jsp.feedbacksuccess_jsp._jspService(HttpServletRequest;HttpServletResponse):void
<input checked="" type="checkbox"/>	org.apache.jsp.feedbacksuccess_jsp:41 email = request. getParameter ("email_addr")
<input checked="" type="checkbox"/>	org.apache.jsp.feedbacksuccess_jsp:46 email .toLowerCase()
<input type="checkbox"/>	org.apache.jsp.feedbacksuccess_jsp:46 sanitzieHtmlWithRegex(email.toLowerCase())
<input checked="" type="checkbox"/>	org.apache.jsp.feedbacksuccess_jsp:46 out.print(sanitzieHtmlWithRegex(email.toLowerCase()))

Issue 14 of 14

Issue ID:	b01794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	add99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void login_jsp:40
Line	40
Source File	login_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.http.HttpSession.getAttribute(String):Object <i>via</i> login_jsp:36
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> login_jsp:40

Issue 14 of 14 - Details

Trace

<input type="checkbox"/>	org.apache.jsp.login_jsp._jspService(HttpServletRequest;HttpServletResponse):void
⋮	
<input checked="" type="checkbox"/>	org.apache.jsp.login_jsp:36 error = request.getSession(1).getAttribute("loginError")
⋮	
<input checked="" type="checkbox"/>	org.apache.jsp.login_jsp:40 out.print(error)

H	Common API Call: CrossSiteScripting.Reflected: Insecure Use of Document.Location
Fix Group ID:	b0d99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
API:	Insecure Use of Document.Location
Notes:	
How to Fix:	Reflected Cross-site Scripting

Issue 1 of 4

Issue ID:	cad99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	b0d99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\disclaimer.htm
Line	19
Source File	src\disclaimer.htm
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of Document.Location
Caller:	src\disclaimer.htm at line 19

Issue 1 of 4 - Details

Call

```
.location.href = sDst;
```

Issue 2 of 4

Issue ID:	cdd99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	b0d99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\disclaimer.htm
Line	35
Source File	src\disclaimer.htm
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of Document.Location
Caller:	src\disclaimer.htm at line 35

Issue 2 of 4 - Details

Call

```
.location.href = "http" + sDst.substring(4);
```

Issue 3 of 4

Issue ID:	d3d99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	b0d99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\static\inside_jobs.htm
Line	14
Source File	src\static\inside_jobs.htm
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of Document.Location
Caller:	src\static\inside_jobs.htm at line 14

Issue 3 of 4 - Details

Call

```
.location.hash == "#alljobs") {  
    document.location.hash = "";
```

Issue 4 of 4

Issue ID:	d6d99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	b0d99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\static\inside_jobs.htm
Line	25
Source File	src\static\inside_jobs.htm
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of Document.Location
Caller:	src\static\inside_jobs.htm at line 25

Issue 4 of 4 - Details

Call

```
.location.href = "index.jsp?content="+jobs[sp[1]][sp[0]];
```

H	Common API Call: CrossSiteScripting.Reflected: Insecure Use of Document.Write
Fix Group ID:	b1d99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
API:	Insecure Use of Document.Write
Notes:	
How to Fix:	Reflected Cross-site Scripting

Issue 1 of 2

Issue ID:	d0d99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	b1d99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\disclaimer.htm
Line	50
Source File	src\disclaimer.htm
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of Document.Write
Caller:	src\disclaimer.htm at line 50

Issue 1 of 2 - Details

Call

```
.write(encodeURIComponent(sDst))
```

Issue 2 of 2

Issue ID:	d9d99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	b1d99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\static\inside_jobs.htm
Line	28
Source File	src\static\inside_jobs.htm
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of Document.Write
Caller:	src\static\inside_jobs.htm at line 28

Issue 2 of 2 - Details

Call

```
.write("<h2 style='color:#ff0000'>We're sorry, but it appears the position for " + sp[0] + " in group " + sp[1] + " is not open anymore</h2>")
```

H Common API Call: CrossSiteScripting.Reflected: Insecure Use of InnerHTML or OuterHTML

Fix Group ID: aed99bc8-ca7f-eb11-85aa-281878e650eb

Status: Noise

Date: 2021-03-08 04:57:37Z

API: Insecure Use of InnerHTML or OuterHTML

Notes:

How to Fix: [Reflected Cross-site Scripting](#)

Issue 1 of 7

Issue ID:	dfd99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	aed99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\util\serverStatusCheck.html
Line	46
Source File	src\util\serverStatusCheck.html
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of InnerHTML or OuterHTML
Caller:	src\util\serverStatusCheck.html at line 46

Issue 1 of 7 - Details

Call

```
.innerHTML=jsonFetchHostName;
```

Issue 2 of 7

Issue ID:	e2d99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	aed99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\util\serverStatusCheck.html
Line	48
Source File	src\util\serverStatusCheck.html
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of InnerHTML or OuterHTML
Caller:	src\util\serverStatusCheck.html at line 48

Issue 2 of 7 - Details

Call

```
.innerHTML=jsonFetchHostStatus;
```

Issue 3 of 7

Issue ID:	e5d99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	aed99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\util\serverStatusCheck.html
Line	67
Source File	src\util\serverStatusCheck.html
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of InnerHTML or OuterHTML
Caller:	src\util\serverStatusCheck.html at line 74

Issue 3 of 7 - Details

Call

```
.innerHTML=sLastHostName;
```

Issue 4 of 7

Issue ID:	e8d99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	aed99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\util\serverStatusCheck.html
Line	55
Source File	src\util\serverStatusCheck.html
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of InnerHTML or OuterHTML
Caller:	src\util\serverStatusCheck.html at line 55

Issue 4 of 7 - Details

Call

```
.innerHTML='The service returned an error. Please be patient while our administrators fix the issue.';
```

Issue 5 of 7

Issue ID:	ebd99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	aed99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\util\serverStatusCheck.html
Line	62
Source File	src\util\serverStatusCheck.html
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of InnerHTML or OuterHTML
Caller:	src\util\serverStatusCheck.html at line 62

Issue 5 of 7 - Details

Call

```
.innerHTML='The service returned an error. The status service appears to not be available';
```

Issue 6 of 7

Issue ID:	eed99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	aed99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\util\serverStatusCheck.html
Line	69
Source File	src\util\serverStatusCheck.html
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of InnerHTML or OuterHTML
Caller:	src\util\serverStatusCheck.html at line 69

Issue 6 of 7 - Details

Call

```
.innerHTML='The service returned a 401 unauthorized error, indicating it was implemented incorrectly';
```

Issue 7 of 7

Issue ID:	f1d99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	aed99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\util\serverStatusCheck.html
Line	76
Source File	src\util\serverStatusCheck.html
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of InnerHTML or OuterHTML
Caller:	src\util\serverStatusCheck.html at line 76

Issue 7 of 7 - Details

Call

```
.innerHTML='The service returned a 302 redirect, indicating it was implemented incorrectly';
```

H Common API Call: CrossSiteScripting.Reflected: Insecure Use of setAttribute

Fix Group ID:	afd99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
API:	Insecure Use of setAttribute
Notes:	
How to Fix:	Reflected Cross-site Scripting

Issue 1 of 8

Issue ID:	f7d99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	afd99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\util\swfobject.js
Line	4
Source File	src\util\swfobject.js
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of setAttribute
Caller:	src\util\swfobject.js at line 4

Issue 1 of 8 - Details

Call

```
.setAttribute("media",X)
```

Issue 2 of 8

Issue ID:	fad99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	afd99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\util\swfobject.js
Line	4
Source File	src\util\swfobject.js
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of setAttribute
Caller:	src\util\swfobject.js at line 4

Issue 2 of 8 - Details

Call

```
.setAttribute(ac,ai[ac])
```

Issue 3 of 8

Issue ID:	fdd99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	afd99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\util\swfobject.js
Line	4
Source File	src\util\swfobject.js
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of setAttribute
Caller:	src\util\swfobject.js at line 4

Issue 3 of 8 - Details

Call

```
.setAttribute("id",X)
```

Issue 4 of 8

Issue ID:	01da9bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	afd99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\util\swfobject.js
Line	4
Source File	src\util\swfobject.js
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of setAttribute
Caller:	src\util\swfobject.js at line 4

Issue 4 of 8 - Details

Call

```
.setAttribute("name",X)
```

Issue 5 of 8

Issue ID:	04da9bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	afd99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\util\swfobject.js
Line	4
Source File	src\util\swfobject.js
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of setAttribute
Caller:	src\util\swfobject.js at line 4

Issue 5 of 8 - Details

Call

```
.setAttribute("type","text/css")
```

Issue 6 of 8

Issue ID:	07da9bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	afd99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\util\swfobject.js
Line	4
Source File	src\util\swfobject.js
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of setAttribute
Caller:	src\util\swfobject.js at line 4

Issue 6 of 8 - Details

Call

```
.setAttribute("type",q)
```

Issue 7 of 8

Issue ID:	0bda9bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	afd99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\util\swfobject.js
Line	4
Source File	src\util\swfobject.js
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of setAttribute
Caller:	src\util\swfobject.js at line 4

Issue 7 of 8 - Details

Call

```
.setAttribute("value",Y)
```

Issue 8 of 8

Issue ID:	f4d99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	afd99bc8-ca7f-eb11-85aa-281878e650eb
Location	src\util\swfobject.js
Line	4
Source File	src\util\swfobject.js
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
API:	Insecure Use of setAttribute
Caller:	src\util\swfobject.js at line 4

Issue 8 of 8 - Details

Call

```
.setAttribute("class",ai[ac])
```

M Common API Call: CrossSiteScripting.Reflected: Potential Issue With Included Script

Fix Group ID: b7d99bc8-ca7f-eb11-85aa-281878e650eb

Status: Noise

Date: 2021-03-08 04:57:37Z

API: Potential Issue With Included Script

Notes:

How to Fix: [Reflected Cross-site Scripting](#)

Issue 1 of 1

Issue ID: dcd99bc8-ca7f-eb11-85aa-281878e650eb

Severity: **Medium**

Status: Noise

Classification: Definitive

Fix Group ID: [b7d99bc8-ca7f-eb11-85aa-281878e650eb](#)

Location: src\static\personal_investments.htm

Line: 27

Source File: src\static\personal_investments.htm

Availability Impact: Partial

Confidentiality Impact: Partial

Integrity Impact: Partial

Date Created: Monday, March 8, 2021

Last Updated: Thursday, March 11, 2021

CWE: 79

API: Potential Issue With Included Script

Caller: src\static\personal_investments.htm at line 27

Issue 1 of 1 - Details

Call

```
http://~~~~~
```

Fix Group ID:	b3d99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
API:	java.io.PrintWriter.write(java.lang.String):void
Notes:	
How to Fix:	Validation.EncodingRequired

Issue 1 of 1

Issue ID:	26da9bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	b3d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.io.PrintWriter.write(String):void SurveyServlet:102
Line	102
Source File	SurveyServlet
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	116
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> SurveyServlet:82
Sink:	java.io.PrintWriter.write(String):void <i>via</i> SurveyServlet:102

Issue 1 of 1 - Details

Trace

```

[ ] com.ibm.security.appscan.altoromutual.servlet.SurveyServlet.doGet(HttpServletRequest;HttpServletResponse):void
[ ] com.ibm.security.appscan.altoromutual.servlet.SurveyServlet:82
  request.getParameter("txtEmail")
[ ] com.ibm.security.appscan.altoromutual.servlet.SurveyServlet:82
  new StringBuilder().append(request.getParameter("txtEmail"))
[ ] com.ibm.security.appscan.altoromutual.servlet.SurveyServlet:82
  new StringBuilder().append(request.getParameter("txtEmail")).append("</b></p></div>")
[ ] com.ibm.security.appscan.altoromutual.servlet.SurveyServlet:81
  content = new StringBuilder().append(request.getParameter("txtEmail")).append("</b></p></div>").toString()
[ ] com.ibm.security.appscan.altoromutual.servlet.SurveyServlet:102
  response.getWriter().write(content)

```

HCommon API Call: Validation.Required:
javax.servlet.http.HttpSession.setAttribute(java.lang.String;ja...

Fix Group ID: b4d99bc8-ca7f-eb11-85aa-281878e650eb

Status: Noise

Date: 2021-03-08 04:57:37Z

API: javax.servlet.http.HttpSession.setAttribute(java.lang.String;java.lang.Object):void

Notes:

How to Fix: [Validation.Required](#)

Issue 1 of 1

Issue ID:	23da9bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	b4d99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.http.HttpSession.setAttribute(String;Object):void SurveyServlet:99
Line	99
Source File	SurveyServlet
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	20
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> SurveyServlet:47
Sink:	javax.servlet.http.HttpSession.setAttribute(String;Object):void <i>via</i> SurveyServlet:99

Issue 1 of 1 - Details

Trace

```

[ ] com.ibm.security.appscan.altoromutual.servlet.SurveyServlet.doGet(HttpServletRequest;HttpServletResponse):void
...
[ ] com.ibm.security.appscan.altoromutual.servlet.SurveyServlet:47
  step = request.getParameter("step")
...
[ ] com.ibm.security.appscan.altoromutual.servlet.SurveyServlet:99
  request.getSession().setAttribute("surveyStep", step)

```

HCommon API Call: Validation.Required:
javax.servlet.ServletRequest.setAttribute(java.lang.String;java...




Fix Group ID:	b2d99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
API:	javax.servlet.ServletRequest.setAttribute(java.lang.String;java.lang.Object):void
Notes:	
How to Fix:	Validation.Required

Issue 1 of 2

Issue ID:	1dda9bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	b2d99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.ServletRequest.setAttribute(String;Object):void FeedbackServlet:49
Line	49
Source File	FeedbackServlet
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	20
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> FeedbackServlet:47
Sink:	javax.servlet.ServletRequest.setAttribute(String;Object):void <i>via</i> FeedbackServlet:49

Issue 1 of 2 - Details

Trace

	com.ibm.security.appscan.altoromutual.servlet.FeedbackServlet.doPost(HttpServletRequest;HttpServletResponse):void
	com.ibm.security.appscan.altoromutual.servlet.FeedbackServlet:47 name = request. getParameter ("name")
	com.ibm.security.appscan.altoromutual.servlet.FeedbackServlet:49 request.setAttribute("message_feedback", name)

Issue 2 of 2

Issue ID:	20da9bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	b2d99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.ServletRequest.setAttribute(String;Object):void SubscribeServlet:50
Line	50
Source File	SubscribeServlet
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	20
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> SubscribeServlet:43
Sink:	javax.servlet.ServletRequest.setAttribute(String;Object):void <i>via</i> SubscribeServlet:50

Issue 2 of 2 - Details

Trace

	com.ibm.security.appscan.altoromutual.servlet.SubscribeServlet.doPost(HttpServletRequest;HttpServletResponse):void
	com.ibm.security.appscan.altoromutual.servlet.SubscribeServlet:43 email = request. getParameter ("txtEmail")
	com.ibm.security.appscan.altoromutual.servlet.SubscribeServlet:50 new StringBuilder ().append(email)
	com.ibm.security.appscan.altoromutual.servlet.SubscribeServlet:50 new StringBuilder ().append(email).append(" has been accepted.")
	com.ibm.security.appscan.altoromutual.servlet.SubscribeServlet:50 new StringBuilder ().append(email).append(" has been accepted.").toString()
	com.ibm.security.appscan.altoromutual.servlet.SubscribeServlet:50 request.setAttribute("message_subscribe", new StringBuilder ().append(email).append(" has been accepted.").toString())

M	Common Open Source: OpenSource: commons-codec-1.6.jar
Fix Group ID:	a1d99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
Library name:	commons-codec-1.6.jar
Notes:	
How to Fix:	Vulnerable Open Source Component See also issue-details 'Resolution' section below.

Issue 1 of 2

Issue ID:	bed99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	Medium
Status	Noise
Classification	Definitive
Fix Group ID:	a1d99bc8-ca7f-eb11-85aa-281878e650eb
Location	.._Altor1940712830\d\stage\dependency\AltoroJ.war.child\WEB-INF\lib\commons-codec-1.6.jar
Source File	.._Altor1940712830\d\stage\dependency\AltoroJ.war.child\WEB-INF\lib\commons-codec-1.6.jar
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CVE	https://issues.apache.org/jira/browse/CODEC-55
CWE:	829
File:	.._Altor1940712830\d\stage\dependency\AltoroJ.war.child\WEB-INF\lib\commons-codec-1.6.jar
Name:	2009-0001
Description:	Not all "business" method implementations of public API in Apache Commons Codec 1.x are thread safe, which might disclose the wrong data or allow an attacker to change non-private fields.Updated 2018-10-07 - an additional review by WhiteSource research team could not indicate on a clear security vulnerability.
URL:	https://issues.apache.org/jira/browse/CODEC-55

Issue 1 of 2 - Details

Name: 2009-0001

Description: Not all "business" method implementations of public API in Apache Commons Codec 1.x are thread safe, which might disclose the wrong data or allow an attacker to change non-private fields.Updated 2018-10-07 - an additional review by WhiteSource research team could not indicate on a clear security vulnerability.

URL: <https://issues.apache.org/jira/browse/CODEC-55>

Issue 2 of 2

Issue ID:	c1d99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	Medium
Status	Noise
Classification	Definitive
Fix Group ID:	a1d99bc8-ca7f-eb11-85aa-281878e650eb
Location	.._Altor1940712830\d\stage\dependency\AltoroJ.war.child\WEB-INF\lib\commons-codec-1.6.jar
Source File	.._Altor1940712830\d\stage\dependency\AltoroJ.war.child\WEB-INF\lib\commons-codec-1.6.jar
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CVE	https://issues.apache.org/jira/browse/CODEC-96
CWE:	829
File:	.._Altor1940712830\d\stage\dependency\AltoroJ.war.child\WEB-INF\lib\commons-codec-1.6.jar
Name:	2010-0001
Description:	Base64 encode() method is no longer thread-safe in Apache Commons Codec before version 1.7, which might disclose the wrong data or allow an attacker to change non-private fields.
Resolution:	Upgrade to version 1.7
URL:	https://issues.apache.org/jira/browse/CODEC-96

Issue 2 of 2 - Details

Name: 2010-0001

Description: Base64 encode() method is no longer thread-safe in Apache Commons Codec before version 1.7, which might disclose the wrong data or allow an attacker to change non-private fields.

Resolution: Upgrade to version 1.7

URL: <https://issues.apache.org/jira/browse/CODEC-96>

M	Common Open Source: OpenSource: derby.jar
Fix Group ID:	a2d99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
Library name:	derby.jar
Notes:	
How to Fix:	Vulnerable Open Source Component See also issue-details 'Resolution' section below.

Issue 1 of 2

Issue ID:	c4d99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	Medium
Status	Noise
Classification	Definitive
Fix Group ID:	a2d99bc8-ca7f-eb11-85aa-281878e650eb
Location	.._Altor1940712830\d\stage\dependency\AltoroJ.war.child\WEB-INF\lib\derby.jar
Source File	.._Altor1940712830\d\stage\dependency\AltoroJ.war.child\WEB-INF\lib\derby.jar
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CVE	https://vuln.whitesourcesoftware.com/vulnerability/CVE-2015-1832
CWE:	829
File:	.._Altor1940712830\d\stage\dependency\AltoroJ.war.child\WEB-INF\lib\derby.jar
Name:	CVE-2015-1832
Description:	XML external entity (XXE) vulnerability in the SqlXmlUtil code in Apache Derby before 10.12.1.1, when a Java Security Manager is not in place, allows context-dependent attackers to read arbitrary files or cause a denial of service (resource consumption) via vectors involving XmlVTI and the XML datatype.
Resolution:	Upgrade to version 10.12.1.1
URL:	https://vuln.whitesourcesoftware.com/vulnerability/CVE-2015-1832

Issue 1 of 2 - Details

Name: CVE-2015-1832

Description: XML external entity (XXE) vulnerability in the SqlXmlUtil code in Apache Derby before 10.12.1.1, when a Java Security Manager is not in place, allows context-dependent attackers to read arbitrary files or cause a denial of service (resource consumption) via vectors involving XmlVTI and the XML datatype.

Resolution: Upgrade to version 10.12.1.1

URL: <https://vuln.whitesourcesoftware.com/vulnerability/CVE-2015-1832>

Issue 2 of 2

Issue ID:	c7d99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	Low
Status	Noise
Classification	Definitive
Fix Group ID:	a2d99bc8-ca7f-eb11-85aa-281878e650eb
Location	.._Altor1940712830\d\stage\dependency\AltoroJ.war.child\WEB-INF\lib\derby.jar
Source File	.._Altor1940712830\d\stage\dependency\AltoroJ.war.child\WEB-INF\lib\derby.jar
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CVE	https://vuln.whitesourcesoftware.com/vulnerability/CVE-2018-1313
CWE:	829
File:	.._Altor1940712830\d\stage\dependency\AltoroJ.war.child\WEB-INF\lib\derby.jar
Name:	CVE-2018-1313
Description:	In Apache Derby 10.3.1.4 to 10.14.1.0, a specially-crafted network packet can be used to request the Derby Network Server to boot a database whose location and contents are under the user's control. If the Derby Network Server is not running with a Java Security Manager policy file, the attack is successful. If the server is using a policy file, the policy file must permit the database location to be read for the attack to work. The default Derby Network Server policy file distributed with the affected releases includes a permissive policy as the default Network Server policy, which allows the attack to work.
Resolution:	Upgrade to version 10.14.2.0
URL:	https://vuln.whitesourcesoftware.com/vulnerability/CVE-2018-1313

Issue 2 of 2 - Details

Name: CVE-2018-1313

Description: In Apache Derby 10.3.1.4 to 10.14.1.0, a specially-crafted network packet can be used to request the Derby Network Server to boot a database whose location and contents are under the user's control. If the Derby Network Server is not running with a Java Security Manager policy file, the attack is successful. If the server is using a policy file, the policy file must permit the database location to be read for the attack to work. The default Derby Network Server policy file distributed with the affected releases includes a permissive policy as the default Network Server policy, which allows the attack to work.

Resolution: Upgrade to version 10.14.2.0

URL: <https://vuln.whitesourcesoftware.com/vulnerability/CVE-2018-1313>

Fix Group ID:	a0d99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
Library name:	swfobject.js
Notes:	
How to Fix:	Vulnerable Open Source Component See also issue-details 'Resolution' section below.

Issue 1 of 1

Issue ID:	bdd99bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a0d99bc8-ca7f-eb11-85aa-281878e650eb
Location	.._Altor1940712830\d\stage\analyze\util\swfobject.js
Source File	.._Altor1940712830\d\stage\analyze\util\swfobject.js
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CVE	https://vuln.whitesourcesoftware.com/vulnerability/CVE-2012-2400
CWE:	829
File:	.._Altor1940712830\d\stage\analyze\util\swfobject.js
Name:	CVE-2012-2400
Description:	Unspecified vulnerability in wp-includes/js/swfobject.js in WordPress before 3.3.2 has unknown impact and attack vectors.
Resolution:	Upgrade to version 3.3.2
URL:	https://vuln.whitesourcesoftware.com/vulnerability/CVE-2012-2400

Issue 1 of 1 - Details

Name:	CVE-2012-2400
Description:	Unspecified vulnerability in wp-includes/js/swfobject.js in WordPress before 3.3.2 has unknown impact and attack vectors.
Resolution:	Upgrade to version 3.3.2
URL:	https://vuln.whitesourcesoftware.com/vulnerability/CVE-2012-2400

H Common Fix Point: Injection.SQL: com.ibm.security.appscan.althoromutual.model.User.getUserTransac...

Fix Group ID:	a5d99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
Location of fix:	com.ibm.security.appscan.altoromutual.model.User.getUserTransactions(java.lang.String;java.lang.String;com.ibm.security.appscan.altoromutual.model.Account[]):com.ibm.security.appscan.altoromutual.model.Transaction[]
Notes:	
How to Fix:	SQL Injection

Issue 1 of 3

Issue ID:	5e1794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a5d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.executeQuery(String):ResultSet DBUtil:317
Line	317
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> transaction_jsp:41
Sink:	java.sql.Statement.executeQuery(String):ResultSet <i>via</i> DBUtil:317

Issue 1 of 3 - Details

Trace

```

org.apache.jsp.bank.transaction_jsp._jspService(HttpServletRequest;HttpServletResponse):void
├── org.apache.jsp.bank.transaction_jsp:41
│   └── endString = request.getParameter("endTime")
├── org.apache.jsp.bank.transaction_jsp:48
│   └── transactions = user.getUserTransactions(startString, endString, user.getAccounts())
│       ├── com.ibm.security.appscan.altoromutual.model.User:96
│       │   └── transactions = getTransactions(startDate, endDate, accounts, -1)
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:310
│       │       │   └── new StringBuilder().append(endDate)
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:310
│       │       │   └── new StringBuilder().append(endDate).append(" 23:59:59")
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:310
│       │       │   └── dateString = new StringBuilder().append(endDate).append(" 23:59:59").toString()
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:313
│       │       │   └── new StringBuilder().append(dateString)
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:313
│       │       │   └── new StringBuilder().append(dateString).append(" ")
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:313
│       │       │   └── new StringBuilder().append(dateString).append(" ").toString()
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:313
│       │       │   └── Temp#18@0 = Temp#18@0.append(Temp#19@0)
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:313
│       │       │   └── Temp#18@0.append(Temp#19@0).append("ORDER BY DATE DESC")
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:313
│       │       │   └── query = Temp#18@0.append(Temp#19@0).append("ORDER BY DATE DESC").toString()
│       │       └── com.ibm.security.appscan.altoromutual.util.DBUtil:317
│       │           └── resultSet = statement.executeQuery(query)

```

Issue 2 of 3

Issue ID:	611794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a5d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.executeQuery(String):ResultSet DBUtil:317
Line	317
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> transaction_jsp:40
Sink:	java.sql.Statement.executeQuery(String):ResultSet <i>via</i> DBUtil:317

Issue 2 of 3 - Details

Trace

```
org.apache.jsp.bank.transaction_jsp._jspService(HttpServletRequest;HttpServletResponse):void
└─ org.apache.jsp.bank.transaction_jsp:40
  startString = request.getParameter("startTime")
└─ org.apache.jsp.bank.transaction_jsp:48
  transactions = user.getUserTransactions(startString, endString, user.getAccounts())
  └─ com.ibm.security.appscan.altoromutual.model.User:96
    transactions = getTransactions(startDate, endDate, accounts, -1)
    └─ com.ibm.security.appscan.altoromutual.util.DBUtil:308
      new StringBuilder().append(startDate)
      └─ com.ibm.security.appscan.altoromutual.util.DBUtil:308
        new StringBuilder().append(startDate).append(" 00:00:00")
        └─ com.ibm.security.appscan.altoromutual.util.DBUtil:308
          dateString = new StringBuilder().append(startDate).append(" 00:00:00").toString()
          └─ com.ibm.security.appscan.altoromutual.util.DBUtil:313
            new StringBuilder().append(dateString)
            └─ com.ibm.security.appscan.altoromutual.util.DBUtil:313
              new StringBuilder().append(dateString).append(" ")
              └─ com.ibm.security.appscan.altoromutual.util.DBUtil:313
                new StringBuilder().append(dateString).append(" ").toString()
                └─ com.ibm.security.appscan.altoromutual.util.DBUtil:313
                  Temp#18@0 = Temp#18@0.append(Temp#19@0)
                  └─ com.ibm.security.appscan.altoromutual.util.DBUtil:313
                    Temp#18@0.append(Temp#19@0).append("ORDER BY DATE DESC")
                    └─ com.ibm.security.appscan.altoromutual.util.DBUtil:313
                      query = Temp#18@0.append(Temp#19@0).append("ORDER BY DATE DESC").toString()
                      └─ com.ibm.security.appscan.altoromutual.util.DBUtil:317
                        resultSet = statement.executeQuery(query)
```

Issue 3 of 3

Issue ID:	641794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a5d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.executeQuery(String):ResultSet DBUtil:317
Line	317
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> transaction_jsp:40
Sink:	java.sql.Statement.executeQuery(String):ResultSet <i>via</i> DBUtil:317

Trace

```

org.apache.jsp.bank.transaction_jsp._jspService(HttpServletRequest;HttpServletResponse):void
└─ org.apache.jsp.bank.transaction_jsp:40
  startString = request.getParameter("startTime")
└─ org.apache.jsp.bank.transaction_jsp:48
  transactions = user.getUserTransactions(startString, endString, user.getAccounts())
  └─ com.ibm.security.appscan.altoromutual.model.User:96
    transactions = getTransactions(startDate, endDate, accounts, -1)
    └─ com.ibm.security.appscan.altoromutual.util.DBUtil:306
      new StringBuilder().append(startDate)
      └─ com.ibm.security.appscan.altoromutual.util.DBUtil:306
        new StringBuilder().append(startDate).append(" 00:00:00' AND ")
        └─ com.ibm.security.appscan.altoromutual.util.DBUtil:306
          new StringBuilder().append(startDate).append(" 00:00:00' AND ").append(endDate)
          └─ com.ibm.security.appscan.altoromutual.util.DBUtil:306
            new StringBuilder().append(startDate).append(" 00:00:00' AND ").append(endDate).append(" 23:59:59")
            └─ com.ibm.security.appscan.altoromutual.util.DBUtil:306
              dateString = new StringBuilder().append(startDate).append(" 00:00:00' AND ").append(endDate).append(" 23:59:59").toString()
              └─ com.ibm.security.appscan.altoromutual.util.DBUtil:313
                new StringBuilder().append(dateString)
                └─ com.ibm.security.appscan.altoromutual.util.DBUtil:313
                  new StringBuilder().append(dateString).append(" ")
                  └─ com.ibm.security.appscan.altoromutual.util.DBUtil:313
                    new StringBuilder().append(dateString).append(" ").toString()
                    └─ com.ibm.security.appscan.altoromutual.util.DBUtil:313
                      Temp#18@0 = Temp#18@0.append(Temp#19@0)
                      └─ com.ibm.security.appscan.altoromutual.util.DBUtil:313
                        Temp#18@0.append(Temp#19@0).append("ORDER BY DATE DESC")
                        └─ com.ibm.security.appscan.altoromutual.util.DBUtil:313
                          query = Temp#18@0.append(Temp#19@0).append("ORDER BY DATE DESC").toString()
                          └─ com.ibm.security.appscan.altoromutual.util.DBUtil:317
                            resultSet = statement.executeQuery(query)
  
```

H	Common Fix Point: Injection.SQL: com.ibm.security.appscan.altoromutual.util.DBUtil.addAccount(ja...
Fix Group ID:	a9d99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
Location of fix:	com.ibm.security.appscan.altoromutual.util.DBUtil.addAccount(java.lang.String;java.lang.String):java.lang.String
Notes:	
How to Fix:	SQL Injection

Issue ID:	671794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a9d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.execute(String):boolean DBUtil:385
Line	385
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> AdminServlet:45
Sink:	java.sql.Statement.execute(String):boolean <i>via</i> DBUtil:385

Issue 1 of 2 - Details

Trace

```

com.ibm.security.appscan.altoromutual.servlet.AdminServlet.doPost(HttpServletRequest;HttpServletResponse):void
├── com.ibm.security.appscan.altoromutual.servlet.AdminServlet:45
│   │ acctType = request.getParameter("accttypes")
│   └── com.ibm.security.appscan.altoromutual.servlet.AdminServlet:49
│       │ error = addAccount(username, acctType)
│       └── com.ibm.security.appscan.altoromutual.util.DBUtil:385
│           │ new StringBuilder().append(username).append(",").append(acctType)
│           ├── com.ibm.security.appscan.altoromutual.util.DBUtil:385
│           │   │ new StringBuilder().append(username).append(",").append(acctType).append(", 0")
│           │   └── com.ibm.security.appscan.altoromutual.util.DBUtil:385
│           │       │ new StringBuilder().append(username).append(",").append(acctType).append(", 0").toString()
│           │       └── com.ibm.security.appscan.altoromutual.util.DBUtil:385
│           │           │ statement.execute(new StringBuilder().append(username).append(",").append(acctType).append(", 0").toString())

```

Issue 2 of 2

Issue ID:	6a1794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a9d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.execute(String):boolean DBUtil:385
Line	385
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> AdminServlet:44
Sink:	java.sql.Statement.execute(String):boolean <i>via</i> DBUtil:385

Issue 2 of 2 - Details

Trace

```

com.ibm.security.appscan.altoromutual.servlet.AdminServlet.doPost(HttpServletRequest;HttpServletResponse):void
├── com.ibm.security.appscan.altoromutual.servlet.AdminServlet:44
│   ├── username = request.getParameter("username")
│   └── com.ibm.security.appscan.altoromutual.servlet.AdminServlet:49
│       ├── error = addAccount(username, acctType)
│       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:385
│       │   ├── new StringBuilder().append(username)
│       │   ├── com.ibm.security.appscan.altoromutual.util.DBUtil:385
│       │   │   ├── new StringBuilder().append(username).append(",")
│       │   │   ├── com.ibm.security.appscan.altoromutual.util.DBUtil:385
│       │   │   │   ├── new StringBuilder().append(username).append(",").append(acctType)
│       │   │   │   ├── com.ibm.security.appscan.altoromutual.util.DBUtil:385
│       │   │   │   │   ├── new StringBuilder().append(username).append(",").append(acctType).append(", 0")
│       │   │   │   │   ├── com.ibm.security.appscan.altoromutual.util.DBUtil:385
│       │   │   │   │   │   ├── new StringBuilder().append(username).append(",").append(acctType).append(", 0").toString()
│       │   │   │   │   │   └── com.ibm.security.appscan.altoromutual.util.DBUtil:385
│       │   │   │   │   │       statement.execute(new StringBuilder().append(username).append(",").append(acctType).append(", 0").toString())

```

H

Common Fix Point: Injection.SQL:

com.ibm.security.appscan.altoromutual.util.DBUtil.addUser(java....

Fix Group ID:	a4d99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
Location of fix:	com.ibm.security.appscan.altoromutual.util.DBUtil.addUser(java.lang.String;java.lang.String;java.lang.String;java.lang.String);java.lang.String
Notes:	
How to Fix:	SQL Injection

Issue 1 of 4

Issue ID:	6d1794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a4d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.execute(String):boolean DBUtil:396
Line	396
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> AdminServlet:58
Sink:	java.sql.Statement.execute(String):boolean <i>via</i> DBUtil:396

Issue 1 of 4 - Details

Trace

<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.servlet.AdminServlet.doPost(HttpServletRequest;HttpServletResponse):void
<input checked="" type="checkbox"/>	com.ibm.security.appscan.altoromutual.servlet.AdminServlet:58 lastname = request. getParameter ("lastname")
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.servlet.AdminServlet:80 error = addUser(username, password1, firstname, lastname)
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.util.DBUtil:396 new StringBuilder ().append(username).append(",").append(password).append(",").append(firstname).append(",").append(lastname)
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.util.DBUtil:396 new StringBuilder ().append(username).append(",").append(password).append(",").append(firstname).append(",").append(lastname).append(",user")
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.util.DBUtil:396 new StringBuilder ().append(username).append(",").append(password).append(",").append(firstname).append(",").append(lastname).append(",user").toString()
<input checked="" type="checkbox"/>	com.ibm.security.appscan.altoromutual.util.DBUtil:396 statement.execute(new StringBuilder ().append(username).append(",").append(password).append(",").append(firstname).append(",").append(lastname).append(",user").toString())

Issue 2 of 4

Issue ID:	701794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a4d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.execute(String):boolean DBUtil:396
Line	396
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> AdminServlet:60
Sink:	java.sql.Statement.execute(String):boolean <i>via</i> DBUtil:396

Issue 2 of 4 - Details

Trace

```
com.ibm.security.appscan.altoromutual.servlet.AdminServlet.doPost(HttpServletRequest;HttpServletResponse):void
├── com.ibm.security.appscan.altoromutual.servlet.AdminServlet:60
│   ├── password1 = request.getParameter("password1")
│   └── com.ibm.security.appscan.altoromutual.servlet.AdminServlet:80
│       ├── error = addUser(username, password1, firstname, lastname)
│       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:396
│       │   ├── new StringBuilder().append(username).append(",").append(password)
│       │   ├── com.ibm.security.appscan.altoromutual.util.DBUtil:396
│       │   │   ├── new StringBuilder().append(username).append(",").append(password).append(", ")
│       │   │   ├── com.ibm.security.appscan.altoromutual.util.DBUtil:396
│       │   │   │   ├── new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname)
│       │   │   │   ├── com.ibm.security.appscan.altoromutual.util.DBUtil:396
│       │   │   │   │   ├── new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname).append(", ")
│       │   │   │   │   ├── com.ibm.security.appscan.altoromutual.util.DBUtil:396
│       │   │   │   │   │   ├── new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname).append(", ").append(lastname)
│       │   │   │   │   │   ├── com.ibm.security.appscan.altoromutual.util.DBUtil:396
│       │   │   │   │   │   │   ├── new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname).append(", ").append(lastname).append(",user")
│       │   │   │   │   │   │   ├── com.ibm.security.appscan.altoromutual.util.DBUtil:396
│       │   │   │   │   │   │   │   ├── new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname).append(", ").append(lastname).append(",user").toString()
│       │   │   │   │   │   │   └── com.ibm.security.appscan.altoromutual.util.DBUtil:396
│       │   │   │   │   │   │   │   ├── statement.execute(new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname).append(", ").append(lastname).append(",user").toString())
```

Issue 3 of 4

Issue ID:	731794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a4d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.execute(String):boolean DBUtil:396
Line	396
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> AdminServlet:57
Sink:	java.sql.Statement.execute(String):boolean <i>via</i> DBUtil:396

Issue 3 of 4 - Details

Trace

```

com.ibm.security.appscan.altoromutual.servlet.AdminServlet.doPost(HttpServletRequest;HttpServletResponse):void
├── com.ibm.security.appscan.altoromutual.servlet.AdminServlet:57
│   └── firstname = request.getParameter("firstname")
├── com.ibm.security.appscan.altoromutual.servlet.AdminServlet:80
│   └── error = addUser(username, password1, firstname, lastname)
│       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:396
│       │   └── new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname)
│       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:396
│       │   └── new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname).append(", ")
│       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:396
│       │   └── new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname).append(", ").append(last
│       │       name)
│       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:396
│       │   └── new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname).append(", ").append(las
│       │       tname).append(",user")
│       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:396
│       │   └── new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname).append(", ").append(las
│       │       tname).append(",user").toString()
│       └── com.ibm.security.appscan.altoromutual.util.DBUtil:396
│           └── statement.execute(new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname).appen
│               d(", ").append(lastname).append(",user").toString())

```

Issue 4 of 4

Issue ID:	761794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a4d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.execute(String):boolean DBUtil:396
Line	396
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> AdminServlet:59
Sink:	java.sql.Statement.execute(String):boolean <i>via</i> DBUtil:396

Issue 4 of 4 - Details

Trace

```

com.ibm.security.appscan.althoromutual.servlet.AdminServlet.doPost(HttpServletRequest;HttpServletResponse):void
├── com.ibm.security.appscan.althoromutual.servlet.AdminServlet:59
│   ├── username = request.getParameter("username")
│   └── com.ibm.security.appscan.althoromutual.servlet.AdminServlet:80
│       ├── error = addUser(username, password1, firstname, lastname)
│       │   ├── com.ibm.security.appscan.althoromutual.util.DBUtil:396
│       │   │   ├── new StringBuilder().append(username)
│       │   │   ├── com.ibm.security.appscan.althoromutual.util.DBUtil:396
│       │   │   │   ├── new StringBuilder().append(username).append(",")
│       │   │   │   ├── com.ibm.security.appscan.althoromutual.util.DBUtil:396
│       │   │   │   │   ├── new StringBuilder().append(username).append(",").append(password)
│       │   │   │   │   ├── com.ibm.security.appscan.althoromutual.util.DBUtil:396
│       │   │   │   │   │   ├── new StringBuilder().append(username).append(",").append(password).append(", ")
│       │   │   │   │   │   ├── com.ibm.security.appscan.althoromutual.util.DBUtil:396
│       │   │   │   │   │   │   ├── new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname)
│       │   │   │   │   │   │   ├── com.ibm.security.appscan.althoromutual.util.DBUtil:396
│       │   │   │   │   │   │   │   ├── new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname).append(", ")
│       │   │   │   │   │   │   │   ├── com.ibm.security.appscan.althoromutual.util.DBUtil:396
│       │   │   │   │   │   │   │   │   ├── new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname).append(", ").append(lastname)
│       │   │   │   │   │   │   │   ├── com.ibm.security.appscan.althoromutual.util.DBUtil:396
│       │   │   │   │   │   │   │   │   ├── new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname).append(", ").append(lastname).append(",user")
│       │   │   │   │   │   │   │   ├── com.ibm.security.appscan.althoromutual.util.DBUtil:396
│       │   │   │   │   │   │   │   │   ├── new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname).append(", ").append(lastname).append(",user").toString()
│       │   │   │   │   │   │   │   ├── com.ibm.security.appscan.althoromutual.util.DBUtil:396
│       │   │   │   │   │   │   │   │   ├── statement.execute(new StringBuilder().append(username).append(",").append(password).append(", ").append(firstname).append(", ").append(lastname).append(",user").toString())

```

H**Common Fix Point: Injection.SQL:**
com.ibm.security.appscan.altoromutual.util.DBUtil.changePasswor...

Fix Group ID:	aad99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
Location of fix:	com.ibm.security.appscan.altoromutual.util.DBUtil.changePassword(java.lang.String;java.lang.String);java.lang.String
Notes:	
How to Fix:	SQL Injection

Issue 1 of 2

Issue ID:	7c1794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	aad99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.execute(String):boolean DBUtil:408
Line	408
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> AdminServlet:90
Sink:	java.sql.Statement.execute(String):boolean <i>via</i> DBUtil:408

Issue 1 of 2 - Details

Trace


```

com.ibm.security.appscan.altoromutual.servlet.AdminServlet.doPost(HttpServletRequest;HttpServletResponse):void
├── com.ibm.security.appscan.altoromutual.servlet.AdminServlet:90
│   ├── username = request.getParameter("username")
│   └── com.ibm.security.appscan.altoromutual.servlet.AdminServlet:103
│       ├── error = changePassword(username, password1)
│       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:408
│       │   ├── new StringBuilder().append(password).append(" WHERE USER_ID = ").append(username)
│       │   ├── com.ibm.security.appscan.altoromutual.util.DBUtil:408
│       │   │   ├── new StringBuilder().append(password).append(" WHERE USER_ID = ").append(username).append("")
│       │   │   ├── com.ibm.security.appscan.altoromutual.util.DBUtil:408
│       │   │   │   ├── new StringBuilder().append(password).append(" WHERE USER_ID = ").append(username).append("").toString()
│       │   │   └── com.ibm.security.appscan.altoromutual.util.DBUtil:408
│       │   │       ├── statement.execute(new StringBuilder().append(password).append(" WHERE USER_ID = ").append(username).append("").toString())
│       │   └──
│       └──
└──

```

Issue 2 of 2

Issue ID:	791794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	aad99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.execute(String):boolean DBUtil:408
Line	408
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> AdminServlet:91
Sink:	java.sql.Statement.execute(String):boolean <i>via</i> DBUtil:408

Issue 2 of 2 - Details

Trace

```

com.ibm.security.appscan.altoromutual.servlet.AdminServlet.doPost(HttpServletRequest;HttpServletResponse):void
├─ com.ibm.security.appscan.altoromutual.servlet.AdminServlet:91
│  password1 = request.getParameter("password1")
├─ com.ibm.security.appscan.altoromutual.servlet.AdminServlet:103
│  error = changePassword(username, password1)
│  └─ com.ibm.security.appscan.altoromutual.util.DBUtil:408
│     new StringBuilder().append(password)
│     └─ com.ibm.security.appscan.altoromutual.util.DBUtil:408
│        new StringBuilder().append(password).append(" WHERE USER_ID = ")
│        └─ com.ibm.security.appscan.altoromutual.util.DBUtil:408
│           new StringBuilder().append(password).append(" WHERE USER_ID = ").append(username)
│           └─ com.ibm.security.appscan.altoromutual.util.DBUtil:408
│              new StringBuilder().append(password).append(" WHERE USER_ID = ").append(username).append("")
│              └─ com.ibm.security.appscan.altoromutual.util.DBUtil:408
│                 new StringBuilder().append(password).append(" WHERE USER_ID = ").append(username).append("").toString()
│                 └─ com.ibm.security.appscan.altoromutual.util.DBUtil:408
│                    statement.execute(new StringBuilder().append(password).append(" WHERE USER_ID = ").append(username).append("").toString())

```

H	Common Fix Point: Injection.SQL: com.ibm.security.appscan.altoromutual.util.DBUtil.isValidUser(j...
Fix Group ID:	abd99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
Location of fix:	com.ibm.security.appscan.altoromutual.util.DBUtil.isValidUser(java.lang.String;java.lang.String):boolean
Notes:	
How to Fix:	SQL Injection

Issue 1 of 2

Issue ID:	2fda9bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	abd99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.executeQuery(String):ResultSet DBUtil:133
Line	133
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> LoginServlet:82
Sink:	java.sql.Statement.executeQuery(String):ResultSet <i>via</i> DBUtil:133

Issue 1 of 2 - Details

Trace

```

com.ibm.security.appscan.altoromutual.servlet.LoginServlet.doPost(HttpServletRequest;HttpServletResponse):void
├── com.ibm.security.appscan.altoromutual.servlet.LoginServlet:82
│   password = request.getParameter("passw")
│   └── com.ibm.security.appscan.altoromutual.servlet.LoginServlet:83
│       password.trim()
│       └── com.ibm.security.appscan.altoromutual.servlet.LoginServlet:83
│           password = password.trim().toLowerCase()
│           └── com.ibm.security.appscan.altoromutual.servlet.LoginServlet:85
│               isValidUser(username, password)
│               ├── com.ibm.security.appscan.altoromutual.util.DBUtil:133
│               │   new StringBuilder().append(user).append(" AND PASSWORD=").append(password)
│               │   └── com.ibm.security.appscan.altoromutual.util.DBUtil:133
│               │       new StringBuilder().append(user).append(" AND PASSWORD=").append(password).append("")
│               │       └── com.ibm.security.appscan.altoromutual.util.DBUtil:133
│               │           new StringBuilder().append(user).append(" AND PASSWORD=").append(password).append("").toString()
│               │           └── com.ibm.security.appscan.altoromutual.util.DBUtil:133
│               │               resultSet = statement.executeQuery(new StringBuilder().append(user).append(" AND PASSWORD=").append(password).append("").toString())

```

Issue 2 of 2

Issue ID:	3d1794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	abd99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.executeQuery(String):ResultSet DBUtil:133
Line	133
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> LoginServlet:78
Sink:	java.sql.Statement.executeQuery(String):ResultSet <i>via</i> DBUtil:133

Issue 2 of 2 - Details

Trace

```

com.ibm.security.appscan.altoromutual.servlet.LoginServlet.doPost(HttpServletRequest;HttpServletResponse):void
├── com.ibm.security.appscan.altoromutual.servlet.LoginServlet:78
│   ├── username = request.getParameter("uid")
│   └── com.ibm.security.appscan.altoromutual.servlet.LoginServlet:85
│       ├── isValidUser(username, password)
│       │   ├── com.ibm.security.appscan.altoromutual.util.DBUtil:133
│       │   │   ├── new StringBuilder().append(user)
│       │   │   ├── com.ibm.security.appscan.altoromutual.util.DBUtil:133
│       │   │   │   ├── new StringBuilder().append(user).append(" AND PASSWORD=")
│       │   │   │   ├── com.ibm.security.appscan.altoromutual.util.DBUtil:133
│       │   │   │   │   ├── new StringBuilder().append(user).append(" AND PASSWORD=").append(password)
│       │   │   │   │   ├── com.ibm.security.appscan.altoromutual.util.DBUtil:133
│       │   │   │   │   │   ├── new StringBuilder().append(user).append(" AND PASSWORD=").append(password).append("")
│       │   │   │   │   │   ├── com.ibm.security.appscan.altoromutual.util.DBUtil:133
│       │   │   │   │   │   │   ├── new StringBuilder().append(user).append(" AND PASSWORD=").append(password).append("").toString()
│       │   │   │   │   │   │   └── com.ibm.security.appscan.altoromutual.util.DBUtil:133
│       │   │   │   │   │   │   ├── resultSet = statement.executeQuery(new StringBuilder().append(user).append(" AND PASSWORD=").append(password).append("").toString())

```

H

Common Fix Point:CrossSiteScripting: java.lang.StringBuilder.toString():java.lang.String

Fix Group ID:	a7d99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
Location of fix:	java.lang.StringBuilder.toString():java.lang.String
Notes:	
How to Fix:	Cross-site Scripting

Issue 1 of 2

Issue ID:	921794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a7d99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void main_jsp:36
Line	36
Source File	main_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.http.HttpSession.getAttribute(String):Object <i>via</i> main_jsp:33
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> main_jsp:36

Issue 1 of 2 - Details

Trace

<input type="checkbox"/>	org.apache.jsp.bank.main_jsp._jspService(HttpServletRequest;HttpServletResponse):void
<input checked="" type="checkbox"/>	org.apache.jsp.bank.main_jsp:33 user = request.getSession(). getAttribute ("user")
<input type="checkbox"/>	org.apache.jsp.bank.main_jsp:36 user .getLastName()
<input type="checkbox"/>	org.apache.jsp.bank.main_jsp:36 new StringBuilder().append(" ").append(user .getLastName())
<input type="checkbox"/>	org.apache.jsp.bank.main_jsp:36 new StringBuilder().append(" ").append(user.getLastName()).toString()
<input checked="" type="checkbox"/>	org.apache.jsp.bank.main_jsp:36 out.print(new StringBuilder().append(" ").append(user.getLastName()).toString())

Issue 2 of 2

Issue ID:	951794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a7d99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void main_jsp:36
Line	36
Source File	main_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	javax.servlet.http.HttpSession.getAttribute(String):Object <i>via</i> main_jsp:33
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> main_jsp:36

Issue 2 of 2 - Details

Trace

```

org.apache.jsp.bank.main_jsp._jspService(HttpServletRequest;HttpServletResponse):void
├── org.apache.jsp.bank.main_jsp:33
│   └── user = request.getSession().getAttribute("user")
├── org.apache.jsp.bank.main_jsp:36
│   └── user.getFirstName()
├── org.apache.jsp.bank.main_jsp:36
│   └── valueOf(user.getFirstName())
├── org.apache.jsp.bank.main_jsp:36
│   └── new StringBuilder(valueOf(user.getFirstName()))
├── org.apache.jsp.bank.main_jsp:36
│   └── new StringBuilder().append(" ")
├── org.apache.jsp.bank.main_jsp:36
│   └── new StringBuilder().append(" ").append(user.getLastName())
├── org.apache.jsp.bank.main_jsp:36
│   └── new StringBuilder().append(" ").append(user.getLastName()).toString()
└── org.apache.jsp.bank.main_jsp:36
    └── out.print(new StringBuilder().append(" ").append(user.getLastName()).toString())

```

H Common Fix Point:CrossSiteScripting:
java.util.ArrayList.toArray(java.lang.Object[]):java.lang.Objec...

Fix Group ID:	a8d99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
Location of fix:	java.util.ArrayList.toArray(java.lang.Object[]):java.lang.Object[]
Notes:	
How to Fix:	Cross-site Scripting

Issue 1 of 2

Issue ID:	831794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a8d99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void admin_jsp:90
Line	90
Source File	admin_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	java.sql.ResultSet.getString(String):String <i>via</i> DBUtil:350
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> admin_jsp:90

Issue 1 of 2 - Details

Trace

org.apache.jsp.admin.admin_jsp._jspService(HttpServletRequest;HttpServletResponse):void
org.apache.jsp.admin.admin_jsp:31 users = getBankUsers()
com.ibm.security.appscan.altoromutual.util.ServletUtil:156 getBankUsernames()
com.ibm.security.appscan.altoromutual.util.DBUtil:350 name = resultSet. getString ("USER_ID")
com.ibm.security.appscan.altoromutual.util.DBUtil:351 users.add(name)
com.ibm.security.appscan.altoromutual.util.DBUtil:354 return users .toArray(new String()[users .size()])
org.apache.jsp.admin.admin_jsp:90 out.print(user)

Issue 2 of 2

Issue ID:	861794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a8d99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.jsp.JspWriter.print(String):void admin_jsp:126
Line	126
Source File	admin_jsp
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	79
Source:	java.sql.ResultSet.getString(String):String <i>via</i> DBUtil:350
Sink:	javax.servlet.jsp.JspWriter.print(String):void <i>via</i> admin_jsp:126

Issue 2 of 2 - Details

Trace

```

org.apache.jsp.admin.admin_jsp._jspService(HttpServletRequest;HttpServletResponse):void
├── org.apache.jsp.admin.admin_jsp:31
│   └── users = getBankUsers()
│       ├── com.ibm.security.appscan.altoromutual.util.ServletUtil:156
│       │   └── getBankUsernames()
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:350
│       │       │   └── name = resultSet.getString("USER_ID")
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:351
│       │       │   └── users.add(name)
│       │       └── com.ibm.security.appscan.altoromutual.util.DBUtil:354
│       │           └── return users.toArray(new String()[users.size()])
│       └── org.apache.jsp.admin.admin_jsp:126
│           └── out.print(user)

```

H	Common Fix Point:Injection.SQL: java.lang.StringBuilder.append(java.lang.String):java.lang.Stri...
Fix Group ID:	a3d99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
Location of fix:	java.lang.StringBuilder.append(java.lang.String):java.lang.StringBuilder
Notes:	
How to Fix:	SQL Injection

Issue 1 of 10

Issue ID:	431794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a3d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.executeQuery(String):ResultSet DBUtil:156
Line	156
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.http.HttpSession.getAttribute(String):Object <i>via</i> TransferServlet:62
Sink:	java.sql.Statement.executeQuery(String):ResultSet <i>via</i> DBUtil:156

Issue 1 of 10 - Details

Trace

<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.servlet.TransferServlet.doPost(HttpServletRequest;HttpServletResponse):void
<input checked="" type="checkbox"/>	com.ibm.security.appscan.altoromutual.servlet.TransferServlet:62 user = request.getSession(). getAttribute ("user")
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.servlet.TransferServlet:66 userName = user .getUsername()
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.servlet.TransferServlet:118 message = transferFunds(userName , creditActId, debitActId, amount)
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.util.DBUtil:216 user = getUserInfo(username)
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.util.DBUtil:156 new StringBuilder().append(username)
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.util.DBUtil:156 new StringBuilder().append(username).append(" ")
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.util.DBUtil:156 new StringBuilder().append(username).append(" ").toString()
<input checked="" type="checkbox"/>	com.ibm.security.appscan.altoromutual.util.DBUtil:156 resultSet = statement.executeQuery(new StringBuilder().append(username).append(" ").toString())

Issue 2 of 10

Issue ID:	461794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a3d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.executeQuery(String):ResultSet DBUtil:190
Line	190
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.http.HttpSession.getAttribute(String):Object <i>via</i> transfer_jsp:32
Sink:	java.sql.Statement.executeQuery(String):ResultSet <i>via</i> DBUtil:190

Issue 2 of 10 - Details

Trace

```

org.apache.jsp.bank.transfer_jsp._jspService(HttpServletRequest;HttpServletResponse):void
├── org.apache.jsp.bank.transfer_jsp:32
│   └── user = request.getSession().getAttribute("user")
├── org.apache.jsp.bank.transfer_jsp:69
│   └── user.getAccounts()
│       ├── com.ibm.security.appscan.altoromutual.model.User:78
│       │   └── getAccounts(this.username)
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │   └── new StringBuilder().append(username)
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │   └── new StringBuilder().append(username).append(" ")
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │   └── new StringBuilder().append(username).append(" ").toString()
│       │       └── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │           └── resultSet = statement.executeQuery(new StringBuilder().append(username).append(" ").toString())

```

Issue 3 of 10

Issue ID:	491794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a3d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.executeQuery(String):ResultSet DBUtil:190
Line	190
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.http.HttpSession.getAttribute(String):Object <i>via</i> balance_jsp:38
Sink:	java.sql.Statement.executeQuery(String):ResultSet <i>via</i> DBUtil:190

Issue 3 of 10 - Details

Trace

```

org.apache.jsp.bank.balance_jsp._jspService(HttpServletRequest;HttpServletResponse):void
├── org.apache.jsp.bank.balance_jsp:38
│   └── user = request.getSession().getAttribute("user")
├── org.apache.jsp.bank.balance_jsp:43
│   └── balance = user.getAccounts()
│       ├── com.ibm.security.appscan.altoromutual.model.User:78
│       │   └── getAccounts(this.username)
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │   └── new StringBuilder().append(username)
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │   └── new StringBuilder().append(username).append(" ")
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │   └── new StringBuilder().append(username).append(" ").toString()
│       │       └── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │           └── resultSet = statement.executeQuery(new StringBuilder().append(username).append(" ").toString())

```

Issue 4 of 10

Issue ID:	4c1794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a3d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.executeQuery(String):ResultSet DBUtil:190
Line	190
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.http.HttpSession.getAttribute(String):Object <i>via</i> transaction_jsp:39
Sink:	java.sql.Statement.executeQuery(String):ResultSet <i>via</i> DBUtil:190

Issue 4 of 10 - Details

Trace

```

org.apache.jsp.bank.transaction_jsp._jspService(HttpServletRequest;HttpServletResponse):void
├── org.apache.jsp.bank.transaction_jsp:39
│   └── user = request.getSession().getAttribute("user")
├── org.apache.jsp.bank.transaction_jsp:48
│   └── user.getAccounts()
│       ├── com.ibm.security.appscan.altoromutual.model.User:78
│       │   └── getAccounts(this.username)
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │   └── new StringBuilder().append(username)
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │   └── new StringBuilder().append(username).append(" ")
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │   └── new StringBuilder().append(username).append(" ").toString()
│       │       └── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │           └── resultSet = statement.executeQuery(new StringBuilder().append(username).append(" ").toString())

```

Issue 5 of 10

Issue ID:	4f1794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a3d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.executeQuery(String):ResultSet DBUtil:190
Line	190
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> LoginServlet:78
Sink:	java.sql.Statement.executeQuery(String):ResultSet <i>via</i> DBUtil:190

Issue 5 of 10 - Details

Trace

<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.servlet.LoginServlet.doPost(HttpServletRequest;HttpServletResponse):void
<input checked="" type="checkbox"/>	com.ibm.security.appscan.altoromutual.servlet.LoginServlet:78 username = request. getParameter("uid")
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.servlet.LoginServlet:98 user = getUserInfo(username)
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.servlet.LoginServlet:99 accounts = user .getAccounts()
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.model.User:78 getAccounts(this.username)
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.util.DBUtil:190 new StringBuilder().append(username)
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.util.DBUtil:190 new StringBuilder().append(username).append(" ")
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.util.DBUtil:190 new StringBuilder().append(username).append(" ").toString()
<input checked="" type="checkbox"/>	com.ibm.security.appscan.altoromutual.util.DBUtil:190 resultSet = statement.executeQuery(new StringBuilder().append(username).append(" ").toString())

Issue 6 of 10

Issue ID:	521794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a3d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.executeQuery(String):ResultSet DBUtil:190
Line	190
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.http.HttpSession.getAttribute(String):Object <i>via</i> transfer_jsp:32
Sink:	java.sql.Statement.executeQuery(String):ResultSet <i>via</i> DBUtil:190

Issue 6 of 10 - Details

Trace

```

org.apache.jsp.bank.transfer_jsp._jspService(HttpServletRequest;HttpServletResponse):void
├── org.apache.jsp.bank.transfer_jsp:32
│   └── user = request.getSession().getAttribute("user")
├── org.apache.jsp.bank.transfer_jsp:81
│   └── user.getAccounts()
│       ├── com.ibm.security.appscan.altoromutual.model.User:78
│       │   └── getAccounts(this.username)
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │   └── new StringBuilder().append(username)
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │   └── new StringBuilder().append(username).append(" ")
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │   └── new StringBuilder().append(username).append(" ").toString()
│       │       └── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │           └── resultSet = statement.executeQuery(new StringBuilder().append(username).append(" ").toString())

```

Issue 7 of 10

Issue ID:	551794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a3d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.executeQuery(String):ResultSet DBUtil:190
Line	190
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.http.HttpSession.getAttribute(String):Object <i>via</i> main_jsp:33
Sink:	java.sql.Statement.executeQuery(String):ResultSet <i>via</i> DBUtil:190

Issue 7 of 10 - Details

Trace

```

org.apache.jsp.bank.main_jsp._jspService(HttpServletRequest;HttpServletResponse):void
├── org.apache.jsp.bank.main_jsp:33
│   └── user = request.getSession().getAttribute("user")
├── org.apache.jsp.bank.main_jsp:50
│   └── user.getAccounts()
│       ├── com.ibm.security.appscan.altoromutual.model.User:78
│       │   └── getAccounts(this.username)
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │   └── new StringBuilder().append(username)
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │   └── new StringBuilder().append(username).append(" ")
│       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │   └── new StringBuilder().append(username).append(" ").toString()
│       │       └── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │           └── resultSet = statement.executeQuery(new StringBuilder().append(username).append(" ").toString())

```

Issue 8 of 10

Issue ID:	581794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a3d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.executeQuery(String):ResultSet DBUtil:190
Line	190
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.http.HttpSession.getAttribute(String):Object <i>via</i> TransferServlet:62
Sink:	java.sql.Statement.executeQuery(String):ResultSet <i>via</i> DBUtil:190

Issue 8 of 10 - Details

Trace

```

[ ] com.ibm.security.appscan.altoromutual.servlet.TransferServlet.doPost(HttpServletRequest;HttpServletResponse):void
  [ ] com.ibm.security.appscan.altoromutual.servlet.TransferServlet:62
    user = request.getSession().getAttribute("user")
  [ ] com.ibm.security.appscan.altoromutual.servlet.TransferServlet:87
    cookieAccounts = user.getAccounts()
    [ ] com.ibm.security.appscan.altoromutual.model.User:78
      getAccounts(this.username)
        [ ] com.ibm.security.appscan.altoromutual.util.DBUtil:190
          new StringBuilder().append(username)
        [ ] com.ibm.security.appscan.altoromutual.util.DBUtil:190
          new StringBuilder().append(username).append(" ")
        [ ] com.ibm.security.appscan.altoromutual.util.DBUtil:190
          new StringBuilder().append(username).append(" ").toString()
        [ ] com.ibm.security.appscan.altoromutual.util.DBUtil:190
          resultSet = statement.executeQuery(new StringBuilder().append(username).append(" ").toString())
  
```

Issue 9 of 10

Issue ID:	5b1794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a3d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.executeQuery(String):ResultSet DBUtil:190
Line	190
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.http.HttpSession.getAttribute(String):Object <i>via</i> TransferServlet:62
Sink:	java.sql.Statement.executeQuery(String):ResultSet <i>via</i> DBUtil:190

Issue 9 of 10 - Details

Trace

```

com.ibm.security.appscan.altoromutual.servlet.TransferServlet.doPost(HttpServletRequest;HttpServletResponse):void
├── com.ibm.security.appscan.altoromutual.servlet.TransferServlet:62
│   └── user = request.getSession().getAttribute("user")
├── com.ibm.security.appscan.altoromutual.servlet.TransferServlet:66
│   └── userName = user.getUsername()
├── com.ibm.security.appscan.altoromutual.servlet.TransferServlet:118
│   └── message = transferFunds(userName, creditActId, debitActId, amount)
│       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:216
│       │   └── user = getUserInfo(username)
│       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:236
│       │   └── userCC = user.getCreditCardNumber()
│       ├── com.ibm.security.appscan.altoromutual.model.User:86
│       │   └── this.getAccounts()
│       │       ├── com.ibm.security.appscan.altoromutual.model.User:78
│       │       │   └── getAccounts(this.username)
│       │       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │       │   └── new StringBuilder().append(username)
│       │       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │       │   └── new StringBuilder().append(username).append(" ")
│       │       │       ├── com.ibm.security.appscan.altoromutual.util.DBUtil:190
│       │       │       │   └── new StringBuilder().append(username).append(" ").toString()
│       │       │       └── resultSet = statement.executeQuery(new StringBuilder().append(username).append(" ").toString())

```

Issue 10 of 10

Issue ID:	401794ce-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a3d99bc8-ca7f-eb11-85aa-281878e650eb
Location	java.sql.Statement.executeQuery(String):ResultSet DBUtil:156
Line	156
Source File	DBUtil
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	89
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> LoginServlet:78
Sink:	java.sql.Statement.executeQuery(String):ResultSet <i>via</i> DBUtil:156

Issue 10 of 10 - Details

Trace

<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.servlet.LoginServlet.doPost(HttpServletRequest;HttpServletResponse):void
<input checked="" type="checkbox"/>	com.ibm.security.appscan.altoromutual.servlet.LoginServlet:78 username = request. getParameter("uid")
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.servlet.LoginServlet:98 user = getUserInfo(username)
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.util.DBUtil:156 new StringBuilder().append(username)
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.util.DBUtil:156 new StringBuilder().append(username).append(" ")
<input type="checkbox"/>	com.ibm.security.appscan.altoromutual.util.DBUtil:156 new StringBuilder().append(username).append(" ").toString()
<input checked="" type="checkbox"/>	com.ibm.security.appscan.altoromutual.util.DBUtil:156 resultSet = statement.executeQuery(new StringBuilder().append(username).append(" ").toString())

H

Common Fix Point:Validation.Required:
javax.servlet.http.HttpSession.setAttribute(java.lang.String;ja...







Fix Group ID:	a6d99bc8-ca7f-eb11-85aa-281878e650eb
Status:	(Mixed)
Date:	2021-03-08 04:57:37Z
Location of fix:	javax.servlet.http.HttpSession.setAttribute(java.lang.String;java.lang.Object):void
Notes:	
How to Fix:	Validation.Required

Issue 1 of 3

Issue ID:	14da9bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	New
Classification	Definitive
Fix Group ID:	a6d99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.http.HttpSession.setAttribute(String;Object):void AdminServlet:118
Line	118
Source File	AdminServlet
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Monday, March 8, 2021
CWE:	20
Source:	java.lang.Throwable.toString():String <i>via</i> DBUtil:399
Sink:	javax.servlet.http.HttpSession.setAttribute(String;Object):void <i>via</i> AdminServlet:118

Issue 1 of 3 - Details

Trace

	com.ibm.security.appscan.altoromutual.servlet.AdminServlet.doPost(HttpServletRequest;HttpServletResponse):void
	com.ibm.security.appscan.altoromutual.servlet.AdminServlet:80 error = addUser (username, password1, firstname, lastname)
	com.ibm.security.appscan.altoromutual.util.DBUtil:399 return e. toString ()
	com.ibm.security.appscan.altoromutual.servlet.AdminServlet:114 new StringBuilder().append(message)
	com.ibm.security.appscan.altoromutual.servlet.AdminServlet:114 message = new StringBuilder().append(message).toString ()
	com.ibm.security.appscan.altoromutual.servlet.AdminServlet:118 request.getSession().setAttribute("message", message)

Issue 2 of 3

Issue ID:	1ada9bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a6d99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.http.HttpSession.setAttribute(String;Object):void AdminServlet:118
Line	118
Source File	AdminServlet
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	20
Source:	java.lang.Throwable.toString():String <i>via</i> DBUtil:388
Sink:	javax.servlet.http.HttpSession.setAttribute(String;Object):void <i>via</i> AdminServlet:118

Issue 2 of 3 - Details

Trace

	com.ibm.security.appscan.altoromutual.servlet.AdminServlet.doPost(HttpServletRequest;HttpServletResponse):void
	com.ibm.security.appscan.altoromutual.servlet.AdminServlet:49 error = addAccount (username, acctType)
	com.ibm.security.appscan.altoromutual.util.DBUtil:388 return e. toString ()
	com.ibm.security.appscan.altoromutual.servlet.AdminServlet:114 new StringBuilder ().append(message)
	com.ibm.security.appscan.altoromutual.servlet.AdminServlet:114 message = new StringBuilder ().append(message).toString()
	com.ibm.security.appscan.altoromutual.servlet.AdminServlet:118 request.getSession().setAttribute("message" , message)

Issue 3 of 3

Issue ID:	17da9bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	a6d99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.http.HttpSession.setAttribute(String;Object):void AdminServlet:118
Line	118
Source File	AdminServlet
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	20
Source:	java.lang.Throwable.toString():String <i>via</i> DBUtil:411
Sink:	javax.servlet.http.HttpSession.setAttribute(String;Object):void <i>via</i> AdminServlet:118

Issue 3 of 3 - Details

Trace

	com.ibm.security.appscan.altoromutual.servlet.AdminServlet.doPost(HttpServletRequest;HttpServletResponse):void
	com.ibm.security.appscan.altoromutual.servlet.AdminServlet:103 error = changePassword (username, password1)
	com.ibm.security.appscan.altoromutual.util.DBUtil:411 return e. toString ()
	com.ibm.security.appscan.altoromutual.servlet.AdminServlet:114 new StringBuilder ().append(message)
	com.ibm.security.appscan.altoromutual.servlet.AdminServlet:114 message = new StringBuilder ().append(message).toString()
	com.ibm.security.appscan.altoromutual.servlet.AdminServlet:118 request.getSession().setAttribute("message", message)

H

Common Fix Point:Validation.Required.URL.Redirect:
javax.servlet.http.HttpServletRequest.getContextPath():java.lan...

Fix Group ID:	acd99bc8-ca7f-eb11-85aa-281878e650eb
Status:	Noise
Date:	2021-03-08 04:57:37Z
Location of fix:	javax.servlet.http.HttpServletRequest.getContextPath():java.lang.String
Notes:	
How to Fix:	Unvalidated Redirect

Issue 1 of 2

Issue ID:	0eda9bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	acd99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.http.HttpServletResponse.sendRedirect(String):void AccountViewServlet:53
Line	53
Source File	AccountViewServlet
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	601
Source:	javax.servlet.http.HttpServlet.doGet(HttpServletRequest;HttpServletResponse):void via AccountViewServlet:65
Sink:	javax.servlet.http.HttpServletResponse.sendRedirect(String):void via AccountViewServlet:53

Issue 1 of 2 - Details

Trace

	com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet.doGet(HttpServletRequest;HttpServletResponse):void
	com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet:65 this.doGet(request, response)
	com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet:53 request.getContextPath()
	com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet:53 valueOf(request.getContextPath())
	com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet:53 new StringBuilder(valueOf(request.getContextPath()))
	com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet:53 new StringBuilder().append("/bank/main.jsp")
	com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet:53 new StringBuilder().append("/bank/main.jsp").toString()
	com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet:53 response.sendRedirect(new StringBuilder().append("/bank/main.jsp").toString())

Issue 2 of 2

Issue ID:	11da9bc8-ca7f-eb11-85aa-281878e650eb
Severity:	High
Status	Noise
Classification	Definitive
Fix Group ID:	acd99bc8-ca7f-eb11-85aa-281878e650eb
Location	javax.servlet.http.HttpServletResponse.sendRedirect(String):void AccountViewServlet:53
Line	53
Source File	AccountViewServlet
Availability Impact	Partial
Confidentiality Impact	Partial
Integrity Impact	Partial
Date Created	Monday, March 8, 2021
Last Updated	Thursday, March 11, 2021
CWE:	601
Source:	javax.servlet.ServletRequest.getParameter(String):String <i>via</i> AccountViewServlet:51
Sink:	javax.servlet.http.HttpServletResponse.sendRedirect(String):void <i>via</i> AccountViewServlet:53

Issue 2 of 2 - Details

Trace

```

com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet.doGet(HttpServletRequest;HttpServletResponse):void
├─ com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet:51
│   accountName = request.getParameter("listAccounts")
├─ com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet:57
│   new StringBuilder().append(accountName)
├─ com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet:57
│   new StringBuilder().append(accountName).toString()
├─ com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet:57
│   dispatcher = request.getRequestDispatcher(new StringBuilder().append(accountName).toString())
├─ com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet:53
│   request.getContextPath()
├─ com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet:53
│   valueOf(request.getContextPath())
├─ com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet:53
│   new StringBuilder(valueOf(request.getContextPath()))
├─ com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet:53
│   new StringBuilder().append("/bank/main.jsp")
├─ com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet:53
│   new StringBuilder().append("/bank/main.jsp").toString()
└─ com.ibm.security.appscan.altoromutual.servlet.AccountViewServlet:53
    response.sendRedirect(new StringBuilder().append("/bank/main.jsp").toString())

```

How to Fix

H Cross-site Scripting

Risk

XSS attacks can expose the user's session cookie, which could lead to the attacker stealing the users' session and breaking into the user's account, which could lead to impersonation of users.

An attacker could modify and view the users' records and perform transactions as those users, and may be able to perform privileged operations on behalf of the user or gain access to any sensitive data belonging to the user. This would be especially dangerous if the user has administrator permissions.

Running a malicious script on the victim's browser could make it possible to install a Trojan horse program or crypto miner, redirect the user to other pages or sites, modify content presentation, and more.

Cause

Cross-site scripting (XSS) vulnerabilities arise when an attacker sends malicious code to the victim's browser, mostly using JavaScript. When a web application uses input from a user, it generates it in the output without filtering it. This way an attacker can insert a malicious script to the input, the script will return in the response and will run on the victim's browser. In particular, sanitization of hazardous characters was not performed correctly on user input.

Exploit Example

The following example shows a script that returns a parameter value in the response. The parameter value is sent to the script using a GET request, and then returned in the response embedded in the HTML.

```
GET /index.aspx?name=JSmith HTTP/1.1
```

```
HTTP/1.1 200 OK
Server: SomeServer
Date: Sun, 01 Jan 2002 00:31:19 GMT
Content-Type: text/html
Accept-Ranges: bytes
Content-Length: 27
```

```
<HTML>
Hello JSmith
</HTML>
```

An attacker might leverage the attack like this. In this case, the JavaScript code will be executed by the browser.

```
GET /index.aspx?name=>"><script>alert('XSS')</script> HTTP/1.1
```



```
HTTP/1.1 200 OK
Server: SomeServer
Date: Sun, 01 Jan 2002 00:31:19 GMT
Content-Type: text/html
Accept-Ranges: bytes
Content-Length: 83

<HTML>
Hello >"><script>alert('XSS')</script>
</HTML>
```

Recommendations

Fully encode all dynamic data from an untrusted source that is inserted into the webpage, to ensure it is treated as literal text and not as a script that could be executed or markup that could be rendered.

Consider the context in which your data will be used, and contextually encode the data as close as possible to the actual output: e.g. HTML encoding for HTML content; HTML Attribute encoding for data output to attribute values; JavaScript encoding for dynamically generated JavaScript. For example, when HTML encoding non-alphanumeric characters into HTML entities, ``<`` and ``>`` would become ``<`` and ``>``.

As an extra defensive measure, validate all external input on the server, regardless of source. Carefully check each input parameter against a rigorous positive specification (allowlist) defining data type; size; range; format; and acceptable values. Regular expressions or framework controls may be useful in some cases, though this is not a replacement for output encoding.

Output encoding and data validation must be done on all untrusted data, wherever it comes from: e.g. form fields, URL parameters, web service arguments, cookies, any data from the network, environment variables, reverse DNS lookups, query results, request headers, URL components, e-mail, files and filenames, databases, and any external systems that provide data to the application. Remember that such inputs may be obtained indirectly through API calls.

For every web page that is returned by the server, explicitly set the ``Content-Type`` HTTP response header. This header value should define a specific character encoding (charset), such as ``ISO-8859-1`` or ``UTF-8``. When an encoding is not specified, the web browser may choose a different encoding by guessing which encoding is actually being used by the web page, which would allow a potential attacker to bypass XSS protections.

Additionally, set the ``httpOnly`` flag on the session cookie, to prevent any XSS exploits from stealing a user's cookie.

Prefer using a framework or standard library that prevents this vulnerability by automatically encoding all dynamic output based on context, or at least that provides constructs that make it easier to avoid.

For every web page that is returned by the server, explicitly set the ``Content-Security-Policy`` HTTP response header, In order to make it significantly more difficult for the attacker to actually exploit the XSS attack.

CWE

79

External References

- [Cross-site Scripting \(XSS\)](#)
- [OWASP XSS Cheat Sheet](#)

[Go to Table of Contents](#)

XSS attacks can expose the user's session cookie, which could lead to the attacker stealing the users' session and breaking into the user's account, which could lead to impersonation of users.

An attacker could modify and view the users' records and perform transactions as those users, and may be able to perform privileged operations on behalf of the user or gain access to any sensitive data belonging to the user. This would be especially dangerous if the user has administrator permissions.

Running a malicious script on the victim's browser could make it possible to install a Trojan horse program or crypto miner, redirect the user to other pages or sites, modify content presentation, and more.

Cause

Cross-site scripting (XSS) vulnerabilities arise when an attacker sends malicious code to the victim's browser, mostly using JavaScript. When a web application uses input from a user, it generates it in the output without filtering it. This way an attacker can insert a malicious script to the input, the script will return in the response and will run on the victim's browser.

In particular, sanitization of hazardous characters was not performed correctly on user input.

In reflected attacks, an attacker tricks an end user into sending request containing malicious code to a vulnerable Web server, which then reflects the attack back to the end user's browser.

The server receives the malicious data directly from the HTTP request and reflects it back in the HTTP response. The most common method of sending malicious content is adding it as a parameter in a URL that is posted publicly or e-mailed directly to the victim. URLs that contain the malicious script constitute the core of many phishing schemes, whereby the convinced victim visits a URL that refers to a vulnerable site. The site then reflects the malicious content back to the victim, and then the content is executed by the victim's browser.

Exploit Example

The following example shows a script that returns a parameter value in the response.

The parameter value is sent to the script using a GET request, and then returned in the response embedded in the HTML.

```
GET /index.aspx?name=JSmith HTTP/1.1
```

```
HTTP/1.1 200 OK
Server: SomeServer
Date: Sun, 01 Jan 2002 00:31:19 GMT
Content-Type: text/html
Accept-Ranges: bytes
Content-Length: 27
```

```
<HTML>
Hello JSmith
</HTML>
```

An attacker might leverage the attack like this. In this case, the JavaScript code will be executed by the browser.

```
GET /index.aspx?name=>"><script>alert('XSS')</script> HTTP/1.1
```

```
HTTP/1.1 200 OK
Server: SomeServer
Date: Sun, 01 Jan 2002 00:31:19 GMT
Content-Type: text/html
Accept-Ranges: bytes
Content-Length: 83
```

```
<HTML>
Hello >"><script>alert('XSS')</script>
</HTML>
```

Recommendations

Fully encode all dynamic data from an untrusted source that is inserted into the webpage, to ensure it is treated as literal text and not as a script that could be executed or markup that could be rendered.

Consider the context in which your data will be used, and contextually encode the data as close as possible to the actual output: e.g. HTML encoding for HTML content; HTML Attribute encoding for data output to attribute values; JavaScript encoding for dynamically generated JavaScript. For example, when HTML encoding non-alphanumeric characters into HTML entities, `<` and `>` would become `<` and `>`.

As an extra defensive measure, validate all external input on the server, regardless of source. Carefully check each input parameter against a rigorous positive specification (allowlist) defining data type; size; range; format; and acceptable values. Regular expressions or framework controls may be useful in some cases, though this is not a replacement for output encoding.

Output encoding and data validation must be done on all untrusted data, wherever it comes from: e.g. form fields, URL parameters, web service arguments, cookies, any data from the network, environment variables, reverse DNS lookups, query results, request headers, URL components, e-mail, files and filenames, databases, and any external systems that provide data to the application. Remember that such inputs may be obtained indirectly through API calls.

For every web page that is returned by the server, explicitly set the `Content-Type` HTTP response header. This header value should define a specific character encoding (charset), such as `ISO-8859-1` or `UTF-8`. When an encoding is not specified, the web browser may choose a different encoding by guessing which encoding is actually being used by the web page, which would allow a potential attacker to bypass XSS protections.

Additionally, set the `httpOnly` flag on the session cookie, to prevent any XSS exploits from stealing a user's cookie.

Prefer using a framework or standard library that prevents this vulnerability by automatically encoding all dynamic output based on context, or at least that provides constructs that make it easier to avoid.

For every web page that is returned by the server, explicitly set the `Content-Security-Policy` HTTP response header, In order to make it significantly more difficult for the attacker to actually exploit the XSS attack.

CWE

79

External References

- [Cross-site Scripting \(XSS\)](#)
- [OWASP XSS Cheat Sheet](#)

[Go to Table of Contents](#)

H Vulnerable Open Source Component

Risk

A vulnerable third party software component may introduce all manner of vulnerabilities into the application

Cause

A vulnerable version of third party software component is installed in the tested application.

Recommendations

Upgrade to the latest version of the third party software component. We highly recommend contacting the vendor of this product to see if a patch or fix has recently been made available.

CWE

External References

- [CERT coordination center](#)
- [Common Vulnerabilities and Exposures \(CVE\)](#)

[Go to Table of Contents](#)

H SQL Injection

Risk

Potential consequences include the loss of:

Confidentiality - Since SQL databases generally hold sensitive data, loss of confidentiality is a frequent problem with SQL injection vulnerabilities.

Authentication - If poor SQL commands are used to check user names and passwords, it may be possible to connect to a system as another user with no previous knowledge of the password.

Authorization - If authorization information is held in a SQL database, it may be possible to change this information through the successful exploitation of a SQL injection vulnerability.

Integrity - Just as it may be possible to read sensitive information, it is also possible to make changes or even delete this information with a SQL injection attack.

Cause

Sanitization of hazardous characters was not performed correctly on user input.

Dynamically generating queries that include unvalidated user input can lead to SQL injection attacks. An attacker can insert SQL commands or modifiers in the user input that can cause the query to behave in an unsafe manner.

Without sufficient validation and encapsulation of user-controllable inputs, the generated SQL query can cause those inputs to be interpreted as SQL instead of ordinary user data. This can be used to alter query logic to bypass security checks, or to insert additional statements that modify the back-end database, possibly including execution of system commands.

SQL payloads can enter the system through any untrusted data, including user input, data previously stored in the database, files, 3rd party APIs, and more.

Recommendations

Ensure that all user input is validated and filtered on the server side, not just to disallow bad characters such as a single quote (') and double quotes ("), but rather to only allow safe characters. Narrowly define the set of safe characters based on the expected value of the parameter in the request.

Use escaping functions on all user input.

Use the lowest privileges that are required to accomplish the necessary tasks.

Use stored procedures or parameterized database calls that do not allow the injection of code.

CWE

- [OWASP - SQL Injection Prevention Cheat Sheet](#)

[Go to Table of Contents](#)

H Validation.EncodingRequired

Cause

The application uses a method that sends data to an external system and that external system is capable of executing code embedded in the data it receives. If the sent data contains malicious scripts, this can lead to malicious scripting attacks and unexpected behavior on the external system. Therefore, it is important to encode the output data appropriately to prevent it from subsequently being interpreted as script code. Although not a common practice, sometimes it may be necessary to encode received data to prevent malicious scripts from getting into the application in the first place.

Typical vulnerable external systems include scripting-enabled clients such as web browsers and database servers. Thus, this finding is primarily applicable to web applications. If content created from unvalidated user input generates dynamic web pages or is stored in data sources, an attacker can introduce malicious scripts, such as JavaScript, into data that other users may later view in a web browser. When viewed, these scripts will be interpreted by the browser and run with the viewing user's security context for communicating with the originating web site. Thus, the attacker's code runs with the same privileges granted to legitimate code sent by the web site.

For example, in a message board application, an attacker can put the following malicious script onto the message board:

```
This is my message <script>alert(document.cookie)</script>
```

When viewed by another user, the victim's browser interprets the code between the <script> tags, resulting in a cross-site scripting attack. In this case, the victim's session cookie appears to the victim in a popup window. In a real attack scenario, a malicious script can send the cookie to the attacker's web site, and then use it for a session hijacking attack.

Non web-based applications can also stage malicious scripting attacks. For example, any application that generates log messages could attack system or security administrators through web-based log viewers. Also, an application that generates email alert messages can attack web-based or other JavaScript-enabled mail clients.

Recommendations

To defend against this kind of attack, the application should encode certain special characters that may be meaningful to other systems, especially for data that will be sent to other systems. For example, it may be useful to apply HTML entity encoding to all data that will be sent to an HTML browser, which may include web pages and log files. HTML entity encoding translates all non-alphanumeric characters into a special character sequence that HTML enabled viewers will not interpret. For example, < and > become < and > respectively. Likewise, if the contents of a free text field are to be stored in a database, one alternative to filtering out special SQL characters (like ' or -) is simply base64 encoding the text before sending it to the database.

Encoded with HTML entity encoding, the above message will become:

```
This is my message &lt;script&gt;alert(document.cookie)&lt;/script&gt;bad
```

When viewed by a browser, the whole message will be interpreted as pure text.

External References

- [Escaping special characters](#)

[Go to Table of Contents](#)

Cause

Input validation is necessary to ensure the integrity of the dynamic data of the application. Validation is useful to protect against cross-site scripting, SQL and command injection, and corrupt application data fields. Even if there are no directly vulnerable uses of a piece of data inside one application, data that is being passed to other applications should be validated to ensure that those applications are not given bad data. Validation, especially for size and metacharacters that might cause string expansion, is even more important when dealing with fixed size, overflowable buffers.

You should validate input from untrusted sources before using it. The untrusted data sources can be HTTP requests or other network traffic, file systems, databases, and any external systems that provide data to the application. In the case of HTTP requests, validate all parts of the request, including headers, form fields, cookies, and URL components that transfer information from the browser to the server side application.

Attackers use unvalidated parameters to target the application's security mechanisms such as authentication and authorization or business logic, and as the primary vector for exercising many other kinds of error, including buffer overflows. If the unvalidated parameters are stored in log files, used in dynamically generated database queries or shell commands, and/or stored in database tables, attackers may also target the server operating system, a database, back-end processing systems, or even log viewing tools. For example, if the application looks up products from the database using an unvalidated productID from HTTP request. This productID can be manipulated using readily available tools to submit SQL injection attacks to the backend database.

```
final String productID = request.getParameter( "productID" );
final String sql = "Select * from Product Where productID = '" + productID + "'";
final Statement statement = connection.createStatement();
final boolean rsReturned = statement.execute(sql);
```

```
char productID[28];
fscanf(fd, "%28s", productID);
char sql[71] = "Select * from Product Where productID = ";
strncat(sql, productID, 28);
strncat(sql, "'", 1);
SQLPrepare(handle, sql, 71);
SQLRETURN ret = SQLEExecute(handle);
```

Note that using dynamically generated SQL queries is another bad practice. Refer to vulnerability type "SQL for more detail.

```
// This class would simply associate parameter names with a data type, plus bounds for
// numeric data or a regular expression for text.
Validator validator = Validator.getInstance( this.getServletContext );
Boolean valid = false;
try
{
    validator.validate( request );
    valid = true;
}
catch ( ValidationException e )
{
    request.getSession().invalidate();
    out.println("Invalid HTTP request");
    out.close();
}
if ( valid )
{
    final String productId = request.getParameter( "productID" );
    final String sql = "Select * from Product where productID= ?";
    final PreparedStatement ps = con.prepareStatement(sql);
    ps.setString(1,productId);
    ps.execute();
}
}
```

```
char productID[28];
fscanf(fd, "%28s", productID);
regex_t productIDValidator;
regcomp(&productIDValidator, "[^a-z]*", REG_EXTENDED);
int matchCount;
regmatch_t matches;
```

```

regexec(productIDValidator, productID, matches, matchCount, 0);
if(0 == matches)
{
    char sql[70] = "Select * from Product Where productID = '?'";
    int sqlLen = 70;
    SQLPrepare(handle, sql, 70);
    SQLBindParameter(handle, 1, SQL_PARAM_INPUT, SQL_C_CHAR, SQL_CHAR, 28, 0, productID, *sqlLen)
    SQLRETURN ret = SQLExecute(handle);
}
else
{
    HandleBadProductID();
}

```

In this instance, a regular expression constrained input to a known set of characters, through a whitelist (rejecting inputs containing anything not in that set), and of a known and limited length. Using a whitelist instead of a blacklist is important, because it can be difficult to anticipate which characters (especially when Unicode is involved) may cause problems, while it is normally easy to determine which characters are legal in a given input field.

Recommendations

The primary recommendation is to validate each input value against a detailed specification of the expectation for that value. This specification should detail characteristics like the character set allowed, length, whether to allow null, minimum value, maximum value, enumerated values, or a specific regular expression. For example, make sure all email fields have the same format. Also, limit name fields and other text fields to an appropriate character set, no special characters, and with expected minimum and maximum sizes. A input pattern violation can be evidence of an attack and should be logged and responded to appropriately.

There are several possible approaches to input validation in an application. The recommendation is to implement the features in a single component that is invoked in a central location. If this is not possible, then enforce a strong policy for the use of a common set of input validation functions.

H Unvalidated Redirect

Risk

Phishing is a social engineering technique where an attacker masquerades as a legitimate entity with which the victim might do business in order to prompt the user to reveal some confidential information (frequently authentication credentials) that can later be used by an attacker. Phishing is essentially a form of information gathering or "fishing" for information.

An attacker may successfully launch a phishing scam and steal user credentials or other sensitive information such as credit card number, social security number etc.

It is possible to redirecting the user to a web page containing malware that could infect the user's computer.

Cause

An http parameter was found to hold a URL value and cause the web application to redirect the request to the specified URL, an attacker can modify the URL value to a malicious site
The web application performs a redirection to an external site.

Exploit Example

The following example shows a URL redirection to untrusted site.
The redir parameter is used to redirect the user to a different page automatically.

```
GET /MyPage.php?redir=/AnotherPage.php HTTP/1.1
```

An attacker might trick the GET parameter used to redirect the user to an external site

```
GET /MyPage.php?redir=https://www.malware.com HTTP/1.1
```

Recommendations

Avoid using redirects and forwards, if using redirects where it possible do not allow the URL as user input for the destination.

If user input can't be avoided, do not pass the URL to this API without first validating and/or encoding the data. Data that a user can modify must be treated as untrusted data.

A unique session ID value should be sent along with the redirect field value. This session ID is then verified before the actual redirect takes place. This ensures that attackers would have a harder time using the redirect field to propagate their malicious activities.

Sanitize input by creating a list of trusted URLs (This should be based on a allow-list)

Force all redirects to first go through a page notifying users that they are going off of your site, with the destination clearly displayed, and have them click a link to confirm.

CWE

601

External References

- [Unvalidated Redirects and Forwards Cheat Sheet](#)

[Go to Table of Contents](#)

M Authentication.Entity

Cause

Authentication.Entity refers to cases in which an application insecurely performs authentication. For example, if username and password information are stored in the source code in clear text, an attacker who gains access to the source code will learn the credentials needed to compromise the database.

Recommendations

Never store passwords in clear text. If the password must be remembered in memory for a short time to perform authentication null out the value as soon as possible to mitigate reading memory blocks to determine the password. In some languages String objects are kept alive throughout the run of the program. In those instances prefer using a char[] array instead of a String object and set every byte to '0' after the password is used for login.

Risk

XSS attacks can expose the user's session cookie, which could lead to the attacker stealing the users' session and breaking into the user's account, which could lead to impersonation of users.

An attacker could modify and view the users' records and perform transactions as those users, and may be able to perform privileged operations on behalf of the user or gain access to any sensitive data belonging to the user. This would be especially dangerous if the user has administrator permissions.

Running a malicious script on the victim's browser could make it possible to install a Trojan horse program or crypto miner, redirect the user to other pages or sites, modify content presentation, and more.

Cause

Cross-site scripting (XSS) vulnerabilities arise when an attacker sends malicious code to the victim's browser, mostly using JavaScript. When a web application uses input from a user, it generates it in the output without filtering it. This way an attacker can insert a malicious script to the input, the script will return in the response and will run on the victim's browser.

In particular, sanitization of hazardous characters was not performed correctly on user input.

In reflected attacks, an attacker tricks an end user into sending request containing malicious code to a vulnerable Web server, which then reflects the attack back to the end user's browser.

The server receives the malicious data directly from the HTTP request and reflects it back in the HTTP response. The most common method of sending malicious content is adding it as a parameter in a URL that is posted publicly or e-mailed directly to the victim. URLs that contain the malicious script constitute the core of many phishing schemes, whereby the convinced victim visits a URL that refers to a vulnerable site. The site then reflects the malicious content back to the victim, and then the content is executed by the victim's browser.

Exploit Example

The following example shows a script that returns a parameter value in the response.

The parameter value is sent to the script using a GET request, and then returned in the response embedded in the HTML.

```
GET /index.aspx?name=JSmith HTTP/1.1
```

```
HTTP/1.1 200 OK
Server: SomeServer
Date: Sun, 01 Jan 2002 00:31:19 GMT
Content-Type: text/html
Accept-Ranges: bytes
Content-Length: 27
```

```
<HTML>
Hello JSmith
</HTML>
```

An attacker might leverage the attack like this. In this case, the JavaScript code will be executed by the browser.

```
GET /index.aspx?name=><script>alert('XSS')</script> HTTP/1.1
```

```
HTTP/1.1 200 OK
Server: SomeServer
Date: Sun, 01 Jan 2002 00:31:19 GMT
Content-Type: text/html
Accept-Ranges: bytes
Content-Length: 83
```

```
<HTML>
Hello ><script>alert('XSS')</script>
</HTML>
```

Recommendations

Fully encode all dynamic data from an untrusted source that is inserted into the webpage, to ensure it is treated as literal text and not as a script that could be executed or markup that could be rendered.

Consider the context in which your data will be used, and contextually encode the data as close as possible to the actual output: e.g. HTML encoding for HTML content; HTML Attribute encoding for data output to attribute values; JavaScript encoding for dynamically generated JavaScript. For example, when HTML encoding non-alphanumeric characters into HTML entities, `<` and `>` would become `<` and `>`.

As an extra defensive measure, validate all external input on the server, regardless of source. Carefully check each input parameter against a rigorous positive specification (allowlist) defining data type; size; range; format; and acceptable values. Regular expressions or framework controls may be useful in some cases, though this is not a replacement for output encoding.

Output encoding and data validation must be done on all untrusted data, wherever it comes from: e.g. form fields, URL parameters, web service arguments, cookies, any data from the network, environment variables, reverse DNS lookups, query results, request headers, URL components, e-mail, files and filenames, databases, and any external systems that provide data to the application. Remember that such inputs may be obtained indirectly through API calls.

For every web page that is returned by the server, explicitly set the `Content-Type` HTTP response header. This header value should define a specific character encoding (charset), such as `ISO-8859-1` or `UTF-8`. When an encoding is not specified, the web browser may choose a different encoding by guessing which encoding is actually being used by the web page, which would allow a potential attacker to bypass XSS protections.

Additionally, set the `httpOnly` flag on the session cookie, to prevent any XSS exploits from stealing a user's cookie.

Prefer using a framework or standard library that prevents this vulnerability by automatically encoding all dynamic output based on context, or at least that provides constructs that make it easier to avoid.

For every web page that is returned by the server, explicitly set the `Content-Security-Policy` HTTP response header, In order to make it significantly more difficult for the attacker to actually exploit the XSS attack.

CWE

79

External References

- [Cross-site Scripting \(XSS\)](#)
- [OWASP XSS Cheat Sheet](#)

[Go to Table of Contents](#)

H Insecure Use of Document.Location

API:Insecure Use of Document.Location

Cause

Content being assigned to the `document.location` variable could be carrying tainted data. Since this data can be further used within the HyperText Markup Language (HTML) of a page, take steps to validate it.

The Document Object Model (DOM) as defined by the World Wide Web Consortium (W3C) supports setting the location of the document, assuming it is well formed.

Content passed into `document.location` should not be exposed to the user in a way that allows for alteration of that content. Exposing such content to the user can lead to a variety of Cross Site Scripting and HTML Injection attacks.

```
<script language="javascript">
...
var inLocation = <%request.getParameter("location")%>;
document.location = inLocation;
...
</script>
```

Recommendations

In the cases where `document.location` carries input originating from an external location, further validation on the input is recommended.

Validation of such content when it is exposed to the user should be performed in the same manner as server-side input validation. Input validation should not be solely performed by client-side JavaScript components exposed to the user. Validation implemented in this manner can be bypassed by means of manipulating the browser and/or the submitted request. Use Server-Side validation or a combination of Client-Side and Server-Side validation.

Furthermore, any unnecessary content should not be attached to `document.location`.

```
<script language="javascript">... var inLocation = <%request.getEncodedParameter("location")%>; document.location = inLocation;... </script>bad
```

External References

- [document.location](#)
- [Cross-site Scripting \(XSS\)](#)
- [OWASP XSS Cheat Sheet](#)

[Go to Table of Contents](#)

H

Reflected Cross-site Scripting

API:Insecure Use of Document.Write

Risk

XSS attacks can expose the user's session cookie, which could lead to the attacker stealing the users' session and breaking into the user's account, which could lead to impersonation of users.

An attacker could modify and view the users' records and perform transactions as those users, and may be able to perform privileged operations on behalf of the user or gain access to any sensitive data belonging to the user. This would be especially dangerous if the user has administrator permissions.

Running a malicious script on the victim's browser could make it possible to install a Trojan horse program or crypto miner, redirect the user to other pages or sites, modify content presentation, and more.

Cause

Cross-site scripting (XSS) vulnerabilities arise when an attacker sends malicious code to the victim's browser, mostly using JavaScript. When a web application uses input from a user, it generates it in the output without filtering it. This way an attacker can insert a malicious script to the input, the script will return in the response and will run on the victim's browser.

In particular, sanitization of hazardous characters was not performed correctly on user input.

In reflected attacks, an attacker tricks an end user into sending request containing malicious code to a vulnerable Web server, which then reflects the attack back to the end user's browser.

The server receives the malicious data directly from the HTTP request and reflects it back in the HTTP response. The most common method of sending malicious content is adding it as a parameter in a URL that is posted publicly or e-mailed directly to the victim. URLs that contain the malicious script constitute the core of many phishing schemes, whereby the convinced victim visits a URL that refers to a vulnerable site. The site then reflects the malicious content back to the victim, and then the content is executed by the victim's browser.

Exploit Example

The following example shows a script that returns a parameter value in the response. The parameter value is sent to the script using a GET request, and then returned in the response embedded in the HTML.

```
GET /index.aspx?name=JSmith HTTP/1.1
```

```
HTTP/1.1 200 OK
Server: SomeServer
Date: Sun, 01 Jan 2002 00:31:19 GMT
Content-Type: text/html
Accept-Ranges: bytes
Content-Length: 27

<HTML>
Hello JSmith
</HTML>
```

An attacker might leverage the attack like this. In this case, the JavaScript code will be executed by the browser.

```
GET /index.aspx?name=>"><script>alert('XSS')</script> HTTP/1.1
```

```
HTTP/1.1 200 OK
Server: SomeServer
Date: Sun, 01 Jan 2002 00:31:19 GMT
Content-Type: text/html
Accept-Ranges: bytes
Content-Length: 83

<HTML>
Hello >"><script>alert('XSS')</script>
</HTML>
```

Recommendations

Fully encode all dynamic data from an untrusted source that is inserted into the webpage, to ensure it is treated as literal text and not as a script that could be executed or markup that could be rendered.

Consider the context in which your data will be used, and contextually encode the data as close as possible to the actual output: e.g. HTML encoding for HTML content; HTML Attribute encoding for data output to attribute values; JavaScript encoding for dynamically generated JavaScript. For example, when HTML encoding non-alphanumeric characters into HTML entities, `<` and `>` would become `<` and `>`.

As an extra defensive measure, validate all external input on the server, regardless of source. Carefully check each input parameter against a rigorous positive specification (allowlist) defining data type; size; range; format; and acceptable values. Regular expressions or framework controls may be useful in some cases, though this is not a replacement for output encoding.

Output encoding and data validation must be done on all untrusted data, wherever it comes from: e.g. form fields, URL parameters, web service arguments, cookies, any data from the network, environment variables, reverse DNS lookups, query results, request headers, URL components, e-mail, files and filenames, databases, and any external systems that provide data to the application. Remember that such inputs may be obtained indirectly through API calls.

For every web page that is returned by the server, explicitly set the `Content-Type` HTTP response header. This header value should define a specific character encoding (charset), such as `ISO-8859-1` or `UTF-8`. When an encoding is not specified, the web browser may choose a different encoding by guessing which encoding is actually being used by the web page, which would allow a potential attacker to bypass XSS protections.

Additionally, set the `httpOnly` flag on the session cookie, to prevent any XSS exploits from stealing a user's cookie.

Prefer using a framework or standard library that prevents this vulnerability by automatically encoding all dynamic output based on context, or at least that provides constructs that make it easier to avoid.

For every web page that is returned by the server, explicitly set the `Content-Security-Policy` HTTP response header, In order to make it significantly more difficult for the attacker to actually exploit the XSS attack.

CWE

External References

- [Cross-site Scripting \(XSS\)](#)
- [OWASP XSS Cheat Sheet](#)

[Go to Table of Contents](#)

M Reflected Cross-site Scripting

API:Potential Issue With Included Script

Risk

XSS attacks can expose the user's session cookie, which could lead to the attacker stealing the users' session and breaking into the user's account, which could lead to impersonation of users.

An attacker could modify and view the users' records and perform transactions as those users, and may be able to perform privileged operations on behalf of the user or gain access to any sensitive data belonging to the user. This would be especially dangerous if the user has administrator permissions.

Running a malicious script on the victim's browser could make it possible to install a Trojan horse program or crypto miner, redirect the user to other pages or sites, modify content presentation, and more.

Cause

Cross-site scripting (XSS) vulnerabilities arise when an attacker sends malicious code to the victim's browser, mostly using JavaScript. When a web application uses input from a user, it generates it in the output without filtering it. This way an attacker can insert a malicious script to the input, the script will return in the response and will run on the victim's browser.

In particular, sanitization of hazardous characters was not performed correctly on user input.

In reflected attacks, an attacker tricks an end user into sending request containing malicious code to a vulnerable Web server, which then reflects the attack back to the end user's browser.

The server receives the malicious data directly from the HTTP request and reflects it back in the HTTP response. The most common method of sending malicious content is adding it as a parameter in a URL that is posted publicly or e-mailed directly to the victim. URLs that contain the malicious script constitute the core of many phishing schemes, whereby the convinced victim visits a URL that refers to a vulnerable site. The site then reflects the malicious content back to the victim, and then the content is executed by the victim's browser.

Exploit Example

The following example shows a script that returns a parameter value in the response.

The parameter value is sent to the script using a GET request, and then returned in the response embedded in the HTML.

```
GET /index.aspx?name=JSmith HTTP/1.1
```

```
HTTP/1.1 200 OK
Server: SomeServer
Date: Sun, 01 Jan 2002 00:31:19 GMT
Content-Type: text/html
Accept-Ranges: bytes
Content-Length: 27
```

```
<HTML>
Hello JSmith
</HTML>
```

An attacker might leverage the attack like this. In this case, the JavaScript code will be executed by the browser.

```
GET /index.aspx?name=>"><script>alert('XSS')</script> HTTP/1.1
```

```
HTTP/1.1 200 OK
Server: SomeServer
Date: Sun, 01 Jan 2002 00:31:19 GMT
Content-Type: text/html
Accept-Ranges: bytes
Content-Length: 83

<HTML>
Hello >"><script>alert('XSS')</script>
</HTML>
```

Recommendations

Fully encode all dynamic data from an untrusted source that is inserted into the webpage, to ensure it is treated as literal text and not as a script that could be executed or markup that could be rendered.

Consider the context in which your data will be used, and contextually encode the data as close as possible to the actual output: e.g. HTML encoding for HTML content; HTML Attribute encoding for data output to attribute values; JavaScript encoding for dynamically generated JavaScript. For example, when HTML encoding non-alphanumeric characters into HTML entities, `<` and `>` would become `<` and `>`.

As an extra defensive measure, validate all external input on the server, regardless of source. Carefully check each input parameter against a rigorous positive specification (allowlist) defining data type; size; range; format; and acceptable values. Regular expressions or framework controls may be useful in some cases, though this is not a replacement for output encoding.

Output encoding and data validation must be done on all untrusted data, wherever it comes from: e.g. form fields, URL parameters, web service arguments, cookies, any data from the network, environment variables, reverse DNS lookups, query results, request headers, URL components, e-mail, files and filenames, databases, and any external systems that provide data to the application. Remember that such inputs may be obtained indirectly through API calls.

For every web page that is returned by the server, explicitly set the `Content-Type` HTTP response header. This header value should define a specific character encoding (charset), such as `ISO-8859-1` or `UTF-8`. When an encoding is not specified, the web browser may choose a different encoding by guessing which encoding is actually being used by the web page, which would allow a potential attacker to bypass XSS protections.

Additionally, set the `httpOnly` flag on the session cookie, to prevent any XSS exploits from stealing a user's cookie.

Prefer using a framework or standard library that prevents this vulnerability by automatically encoding all dynamic output based on context, or at least that provides constructs that make it easier to avoid.

For every web page that is returned by the server, explicitly set the `Content-Security-Policy` HTTP response header, In order to make it significantly more difficult for the attacker to actually exploit the XSS attack.

CWE

79

External References

- [Cross-site Scripting \(XSS\)](#)
- [OWASP XSS Cheat Sheet](#)

[Go to Table of Contents](#)

Risk

XSS attacks can expose the user's session cookie, which could lead to the attacker stealing the users' session and breaking into the user's account, which could lead to impersonation of users.

An attacker could modify and view the users' records and perform transactions as those users, and may be able to perform privileged operations on behalf of the user or gain access to any sensitive data belonging to the user. This would be especially dangerous if the user has administrator permissions.

Running a malicious script on the victim's browser could make it possible to install a Trojan horse program or crypto miner, redirect the user to other pages or sites, modify content presentation, and more.

Cause

Cross-site scripting (XSS) vulnerabilities arise when an attacker sends malicious code to the victim's browser, mostly using JavaScript. When a web application uses input from a user, it generates it in the output without filtering it. This way an attacker can insert a malicious script to the input, the script will return in the response and will run on the victim's browser.

In particular, sanitization of hazardous characters was not performed correctly on user input.

In reflected attacks, an attacker tricks an end user into sending request containing malicious code to a vulnerable Web server, which then reflects the attack back to the end user's browser.

The server receives the malicious data directly from the HTTP request and reflects it back in the HTTP response. The most common method of sending malicious content is adding it as a parameter in a URL that is posted publicly or e-mailed directly to the victim. URLs that contain the malicious script constitute the core of many phishing schemes, whereby the convinced victim visits a URL that refers to a vulnerable site. The site then reflects the malicious content back to the victim, and then the content is executed by the victim's browser.

Exploit Example

The following example shows a script that returns a parameter value in the response.

The parameter value is sent to the script using a GET request, and then returned in the response embedded in the HTML.

```
GET /index.aspx?name=JSmith HTTP/1.1
```

```
HTTP/1.1 200 OK
Server: SomeServer
Date: Sun, 01 Jan 2002 00:31:19 GMT
Content-Type: text/html
Accept-Ranges: bytes
Content-Length: 27
```

```
<HTML>
Hello JSmith
</HTML>
```

An attacker might leverage the attack like this. In this case, the JavaScript code will be executed by the browser.

```
GET /index.aspx?name=>"><script>alert('XSS')</script> HTTP/1.1
```

```
HTTP/1.1 200 OK
Server: SomeServer
Date: Sun, 01 Jan 2002 00:31:19 GMT
Content-Type: text/html
Accept-Ranges: bytes
Content-Length: 83
```

```
<HTML>
Hello >"><script>alert('XSS')</script>
</HTML>
```

Recommendations

Fully encode all dynamic data from an untrusted source that is inserted into the webpage, to ensure it is treated as literal text and not as a script that could be executed or markup that could be rendered.

Consider the context in which your data will be used, and contextually encode the data as close as possible to the actual output: e.g. HTML encoding for HTML content; HTML Attribute encoding for data output to attribute values; JavaScript encoding for dynamically generated JavaScript. For example, when HTML encoding non-alphanumeric characters into HTML entities, `<` and `>` would become `<` and `>`.

As an extra defensive measure, validate all external input on the server, regardless of source. Carefully check each input parameter against a rigorous positive specification (allowlist) defining data type; size; range; format; and acceptable values. Regular expressions or framework controls may be useful in some cases, though this is not a replacement for output encoding.

Output encoding and data validation must be done on all untrusted data, wherever it comes from: e.g. form fields, URL parameters, web service arguments, cookies, any data from the network, environment variables, reverse DNS lookups, query results, request headers, URL components, e-mail, files and filenames, databases, and any external systems that provide data to the application. Remember that such inputs may be obtained indirectly through API calls.

For every web page that is returned by the server, explicitly set the `Content-Type` HTTP response header. This header value should define a specific character encoding (charset), such as `ISO-8859-1` or `UTF-8`. When an encoding is not specified, the web browser may choose a different encoding by guessing which encoding is actually being used by the web page, which would allow a potential attacker to bypass XSS protections.

Additionally, set the `httpOnly` flag on the session cookie, to prevent any XSS exploits from stealing a user's cookie.

Prefer using a framework or standard library that prevents this vulnerability by automatically encoding all dynamic output based on context, or at least that provides constructs that make it easier to avoid.

For every web page that is returned by the server, explicitly set the `Content-Security-Policy` HTTP response header, In order to make it significantly more difficult for the attacker to actually exploit the XSS attack.

CWE

79

External References

- [Cross-site Scripting \(XSS\)](#)
- [OWASP XSS Cheat Sheet](#)

[Go to Table of Contents](#)

M Authentication.Entity

API:java.sql.DriverManager.getConnection(String):Connection

Cause

Authentication.Entity refers to cases in which an application insecurely performs authentication. For example, if username and password information are stored in the source code in clear text, an attacker who gains access to the source code will learn the credentials needed to compromise the database.

Recommendations

Never store passwords in clear text. If the password must be remembered in memory for a short time to perform authentication null out the value as soon as possible to mitigate reading memory blocks to determine the password. In some languages String objects are kept alive throughout the run of the program. In those instances prefer using a char[] array instead of a String object and set every byte to '0' after the password is used for login.