VERACODE

Veracode Detailed Report

Application Security Report
As of 4 Apr 2018

Prepared for:	Moody's Corporation
Prepared on:	April 5, 2018
Application:	BEAT Flow B
Sandbox:	Sameer Goyal's Sandbox
Industry:	Finance
Business Criticality:	BC3 (Medium)
Required Analysis:	Static, Dynamic
Type(s) of Analysis Conducted:	Static
Scope of Static Scan:	1 of 9 Modules Analyzed

Inside This Report

65 Network Drive, Burlington, MA 01803

Tel.+1.339.674.2500 **Fax.**+1.339.674.2502 **URL:**http://www.veracode.com

Veracode Detailed Report

Application Security Report As of 4 Apr 2018

Veracode Level: VL2

Rated: Apr 4, 2018

Application:	BEAT Flow B	Business Criticality:	Medium
Target Level:	None	Published Rating:	A

Scans Included in Report

Static Scan	Dynamic Scan	Manual Scan
4 Apr 2018 Static Score: 77 Completed: 4/4/18	Not Included in Report	Not Included in Report

Executive Summary

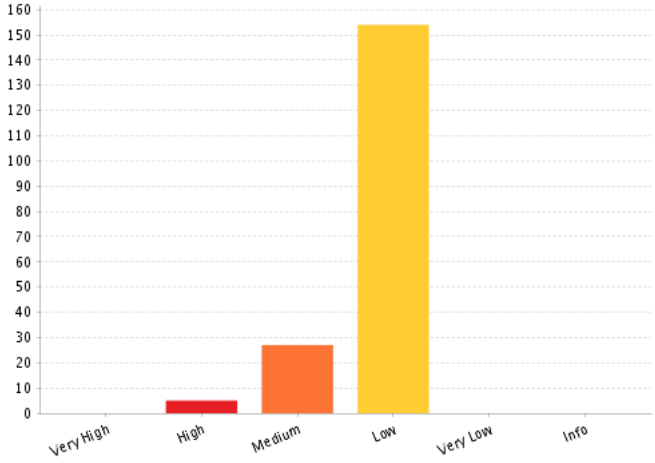
This report contains a summary of the security flaws identified in the application using automated static, automated dynamic and/or manual security analysis techniques. This is useful for understanding the overall security quality of an individual application or for comparisons between applications.

Application Business Criticality: BC3 (Medium)

Impacts:Operational Risk (Low), Financial Loss (Medium)

An application's business criticality is determined by business risk factors such as: reputation damage, financial loss, operational risk, sensitive information disclosure, personal safety, and legal violations. The Veracode Level and required assessment techniques are selected based on the policy assigned to the application.

Summary of Flaws Found by Severity



Analyses Performed vs. Required

	Any	Static	Dynamic	Manual
Performed:		●	○	○
Required:	○	●	●	○

Action Items:

Veracode recommends the following approaches ranging from the most basic to the strong security measures that a vendor can undertake to increase the overall security level of the application.

Required Analysis

- Your policy requires Dynamic Scan but it has not been performed. Please submit your application for Dynamic Scan and remediate the required detected flaws to conform to your assigned policy.
- Your policy requires periodic Static Scan. Your next analysis must be completed by 5/4/18. Please submit your application for Static Scan by the deadline and remediate the required detected flaws to conform to your assigned policy.

Flaws To Fix For Minimum Score

- A rule in your policy requires a minimum score of 80. Fix 4 High flaws to reach a score of 80.

Flaw Severities

- Medium severity flaws and above must be fixed for policy compliance.

65 Network Drive, Burlington, MA 01803

Tel.+1.339.674.2500 **Fax.**+1.339.674.2502 **URL:**http://www.veracode.com

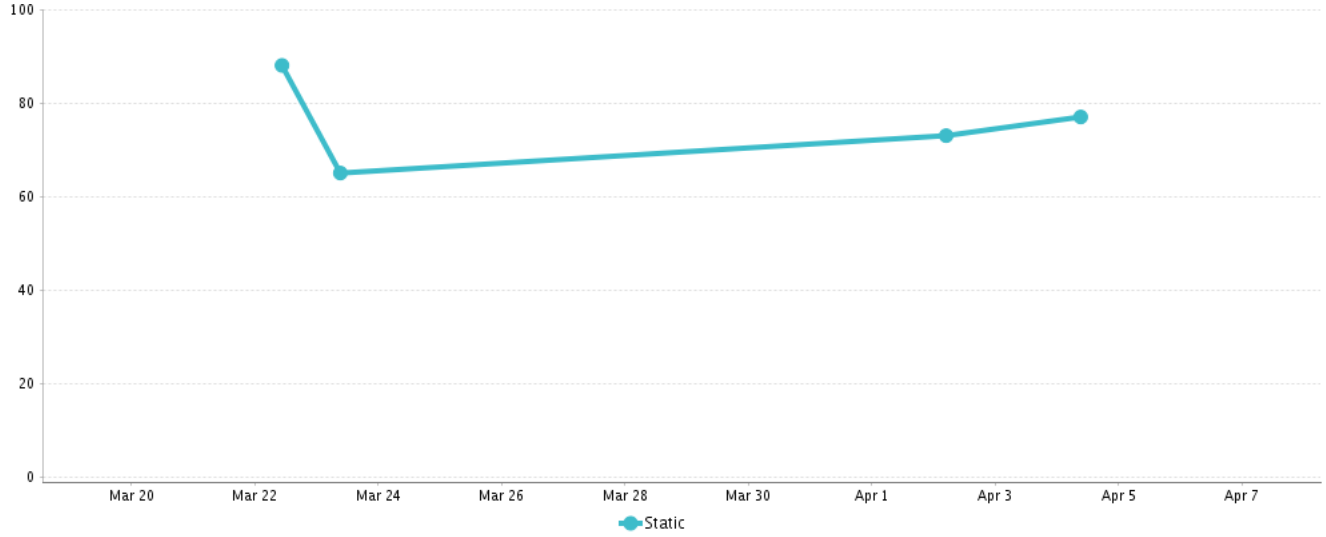
Longer Timeframe (6 - 12 months)

- Certify that software engineers have been trained on application security principles and practices.

65 Network Drive, Burlington, MA 01803

Tel.+1.339.674.2500 **Fax.**+1.339.674.2502 **URL:**http://www.veracode.com

Application Trend Data



Scope of Static Scan

The following modules were included in the static scan because the scan submitter selected them as entry points, which are modules that accept external data.

Engine Version: 120896

The following modules were included in the application scan:

Module Name	Compiler	Operating Environment	Engine Version
App_Web__attachmentdownloaddetails.cshtml.50f944d4.dll	MSIL_MSVC11_X86	Win32	120896

File Differences Between Scans

The uploaded modules for this scan do not match the modules you uploaded for the previous scan. This disparity can affect the scan results even if Veracode did not find flaws in the files with differences. See appendix for more details.

The following modules were not selected for a full scan. Code paths in these modules that are not called from a scanned module are not included in this report.

Module Name	Compiler	Operating Environment	Engine Version
CaptchaMvc.dll	MSIL_MSVC11_X86	Win32	120896
DocumentFormat.OpenXml.dll	MSIL_MSVC14_X86	Win32	120896
itextsharp.dll	MSIL_MSVC8_X86	Win32	120896
Microsoft.AspNet.SignalR.SystemWeb.dll	MSIL_MSVC14_X86	Win32	120896
Workfluxor.Bussiness.dll	MSIL_MSVC11_X86	Win32	120896
Workfluxor.dll	MSIL_MSVC11_X86	Win32	120896
Workfluxor.DocSearchSearvices.dll	MSIL_MSVC11_X86	Win32	120896
Workfluxor.Services.dll	MSIL_MSVC11_X86	Win32	120896

65 Network Drive, Burlington, MA 01803

Tel.+1.339.674.2500 **Fax.**+1.339.674.2502 **URL:**http://www.veracode.com

Flaw Types by Severity and Category

Static Scan Security Quality Score = 77 (+4) from prior scan				
Very High	**0**			
High	**5**	**(-1)**		
Credentials Management		(-1)		
SQL Injection	1			
Untrusted Initialization	4			
Medium	**27**	**(+4)**		
Cross-Site Scripting	4	(+4)		
Directory Traversal	18			
Insufficient Input Validation	5			
Low	**154**	**(-264)**		
Code Quality		(-1)		
Insufficient Input Validation	154	(-263)		
Very Low	**0**			
Informational	**0**			
Total	**186**	**(-261)**		

65 Network Drive, Burlington, MA 01803

Tel.+1.339.674.2500 **Fax.**+1.339.674.2502 **URL:**http://www.veracode.com

Policy Evaluation

Policy Name: MCO - Internet-accessible / Customer-deployed

Revision: 10

Policy Status: Not Assessed

Description

Default policy applied to all Moody's Internet-accessible and customer-deployed applications

Rules

Rule type	Requirement	Findings	Status
Standard	Latest OWASP	Flaws found: 5	Did not pass
Standard	SANS Top 25	Flaws found: 23	Did not pass
Standard	CERT	Flaws found: 18	Did not pass
Max Severity	Medium	Flaws found: 32	Did not pass
Min Analysis Score	80	77	Did not pass

Software Composition Analysis Rules

Rule type	Requirement	Findings	Status
Disallow Component Blacklist	Prevent an application from passing policy if blacklisted components are detected	0 Blacklisted	Passed
Disallow Vulnerabilities by Severity	High and Above Not Allowed	0 Components	Passed

Policy Standards

The table(s) below list the standards in your policy that the application failed to meet. Portions of the standard that had no findings have been suppressed for clarity.

Latest OWASP

Section	Flaw Count
OWASP Top Ten 2013 Category A1 - Injection	1
OWASP Top Ten 2013 Category A3 - Cross-Site Scripting (XSS)	4

SANS Top 25

Section	Flaw Count
Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')	18
Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')	4
2011 Top 25 - Insecure Interaction Between Components	1

Unsupported Frameworks

This report may have incomplete results based on the following unsupported frameworks identified during the static scan:

* ELMAH

The lack of support for all frameworks in use by this application and/or its supporting libraries may prevent the static discovery of some flaws in the application, however, it does not invalidate the flaws that were found.

Findings & Recommendations

Best Practice Findings

You are doing a good job at protecting against these flaw types:

Cross-Site Scripting

CWE–80: Improper Neutralization of Script-Related HTML Tags in a Web Page (Basic XSS)

* This application has 22 opportunities for this flaw, and 18 were successfully defended against using security best practices.
* The remaining 4 flaws should be addressed and are described in the following section, “Detailed Flaws by Severity.”

Detailed Flaws by Severity

Very High (0 flaws)

No flaws of this type were found

High (5 flaws)

Fix Required by Policy

SQL Injection(1 flaw)

Description

SQL injection vulnerabilities occur when data enters an application from an untrusted source and is used to dynamically construct a SQL query. This allows an attacker to manipulate database queries in order to access, modify, or delete arbitrary data. Depending on the platform, database type, and configuration, it may also be possible to execute administrative operations on the database, access the filesystem, or execute arbitrary system commands. SQL injection attacks can also be used to subvert authentication and authorization schemes, which would enable an attacker to gain privileged access to restricted portions of the application.

Recommendations

Several techniques can be used to prevent SQL injection attacks. These techniques complement each other and address security at different points in the application. Using multiple techniques provides defense-in-depth and minimizes the likelihood of a SQL injection vulnerability.

* Use parameterized prepared statements rather than dynamically constructing SQL queries. This will prevent the database from interpreting the contents of bind variables as part of the query and is the most effective defense against SQL injection.
* Validate user-supplied input using positive filters (white lists) to ensure that it conforms to the expected format, using centralized data validation routines when possible.
* Normalize all user-supplied data before applying filters or regular expressions, or submitting the data to a database. This means that all URL-encoded (%xx), HTML-encoded (&#xx;), or other encoding schemes should be reduced to the internal character representation expected by the application. This prevents attackers from using alternate encoding schemes to bypass filters.
* When using database abstraction libraries such as Hibernate, do not assume that all methods exposed by the API will automatically prevent SQL injection attacks. Most libraries contain methods that pass arbitrary queries to the database in an unsafe manner.

Associated Flaws by CWE ID:

Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection') (CWE ID 89)(1 flaw)

Fix Required by Policy

Description

This database query contains a SQL injection flaw. The function call constructs a dynamic SQL query using a variable derived from untrusted input. An attacker could exploit this flaw to execute arbitrary SQL queries against the database.

Effort to Fix: 3 - Complex implementation error. Fix is approx. 51-500 lines of code. Up to 5 days to fix.

Recommendations

Avoid dynamically constructing SQL queries. Instead, use parameterized prepared statements to prevent the database from interpreting the contents of bind variables as part of the query. Always validate untrusted input to ensure that it conforms to the expected format, using centralized data validation routines when possible.

Instances found via Static Scan

Flaw Id	Module #	Class #	Module	Location	Fix By
276	19	-	workfluxor.dll	projects/.../helpers/utility.cs 142	5/7/18

Untrusted Initialization(4 flaws)

Description

Applications should be reluctant to trust variables that have been initialized outside of its trust boundary. Untrusted initialization refers to instances in which an application allows external control of system settings or variables, which can disrupt service or cause an application to behave in unexpected ways. For example, if an application uses values from the environment, assuming the data cannot be tampered with, it may use that data in a dangerous way.

Recommendations

Compartmentalize the application and determine where the trust boundaries exist, then treat any input or control outside the trust boundary as potentially hostile. In general, do not allow user-provided or otherwise untrusted data to control sensitive values.

Associated Flaws by CWE ID:

External Control of System or Configuration Setting (CWE ID 15)(4 flaws)

Description

This call to allows external control of system settings. The argument to the function is constructed using untrusted input, which can disrupt service or cause an application to behave in unexpected ways.

Effort to Fix: 2 - Implementation error. Fix is approx. 6-50 lines of code. 1 day to fix.

Recommendations

Never allow untrusted data to control system-level settings. Always validate untrusted input to ensure that it conforms to the expected format, using centralized data validation routines when possible.

Instances found via Static Scan

Flaw Id	Module #	Class #	Module	Location	Fix By
162	19	-	workfluxor.dll	projects/.../helpers/utility.cs 86	5/7/18
308	19	-	workfluxor.dll	projects/.../helpers/utility.cs 129	5/7/18
414	19	-	workfluxor.dll	projects/.../helpers/utility.cs 173	5/7/18
193	19	-	workfluxor.dll	projects/.../helpers/utility.cs 229	5/7/18

Medium (27 flaws)

Fix Required by Policy

Cross-Site Scripting(4 flaws)

Description

Cross-site scripting (XSS) attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed occur whenever a web application uses untrusted data in the output it generates without validating or encoding it. XSS vulnerabilities are commonly exploited to steal or manipulate cookies, modify presentation of content, and compromise sensitive information, with new attack vectors being discovered on a regular basis. XSS is also commonly referred to as HTML injection.

XSS vulnerabilities can be either persistent or transient (often referred to as stored and reflected, respectively). In a persistent XSS vulnerability, the injected code is stored by the application, for example within a blog comment or message board. The attack occurs whenever a victim views the page containing the malicious script. In a transient XSS vulnerability, the injected code is included directly in the HTTP request. These attacks are often carried out via malicious URLs sent via email or another website and requires the victim to browse to that link. The consequence of an XSS attack to a victim is the same regardless of whether it is persistent or transient; however, persistent XSS vulnerabilities are likely to affect a greater number of victims due to its delivery mechanism.

Recommendations

Several techniques can be used to prevent XSS attacks. These techniques complement each other and address security at different points in the application. Using multiple techniques provides defense-in-depth and minimizes the likelihood of a XSS vulnerability.

* Use output filtering to sanitize all output generated from user-supplied input, selecting the appropriate method of encoding based on the use case of the untrusted data. For example, if the data is being written to the body of an HTML page, use HTML entity encoding. However, if the data is being used to construct generated Javascript or if it is consumed by client-side methods that may interpret it as code (a common technique in Web 2.0 applications), additional restrictions may be necessary beyond simple HTML encoding.
* Validate user-supplied input using positive filters (white lists) to ensure that it conforms to the expected format, using centralized data validation routines when possible.
* Do not permit users to include HTML content in posts, notes, or other data that will be displayed by the application. If users are permitted to include HTML tags, then carefully limit access to specific elements or attributes, and use strict validation filters to prevent abuse.

Associated Flaws by CWE ID:

Improper Neutralization of Script-Related HTML Tags in a Web Page (Basic XSS) (CWE ID 80)(4 flaws)

Fix Required by Policy

Description

This call contains a cross-site scripting (XSS) flaw. The application populates the HTTP response with untrusted input, allowing an attacker to embed malicious content, such as Javascript code, which will be executed in the context of the victim's browser. XSS vulnerabilities are commonly exploited to steal or manipulate cookies, modify presentation of content, and compromise confidential information, with new attack vectors being discovered on a regular basis.

Effort to Fix: 3 - Complex implementation error. Fix is approx. 51-500 lines of code. Up to 5 days to fix.

Recommendations

Use contextual escaping on all untrusted data before using it to construct any portion of an HTTP response. The escaping method should be chosen based on the specific use case of the untrusted data, otherwise it may not protect fully against the attack. For example, if the data is being written to the body of an HTML page, use HTML entity escaping; if the data is being written to an attribute, use attribute escaping; etc. When a web framework provides built-in support for automatic XSS escaping, do not disable it. Both the OWASP Java Encoder library for Java and the Microsoft AntiXSS library provide contextual escaping methods. For more details on contextual escaping, see https://www.owasp.org/index.php/XSS_%%28Cross_Site_Scripting%%29_Prevention_Cheat_Sheet. In addition, as a best practice, always validate untrusted input to ensure that it conforms to the expected format, using centralized data validation routines when possible.

Instances found via Static Scan

	Flaw Id	Module #	Class #	Module	Location	Fix By
NEW	804	17	-	app_web_updaterequest.cshtml.ce0b5a11.dll	.../request/updaterequest.cshtml 755	7/3/18
NEW	807	17	-	app_web_updaterequest.cshtml.ce0b5a11.dll	.../request/updaterequest.cshtml 792	7/3/18
NEW	803	17	-	app_web_updaterequest.cshtml.ce0b5a11.dll	.../request/updaterequest.cshtml 802	7/3/18
NEW	806	17	-	app_web_updaterequest.cshtml.ce0b5a11.dll	.../request/updaterequest.cshtml 810	7/3/18

Directory Traversal(18 flaws)

Description

Allowing user input to control paths used in filesystem operations may enable an attacker to access or modify otherwise protected system resources that would normally be inaccessible to end users. In some cases, the user-provided input may be passed directly to the filesystem operation, or it may be concatenated to one or more fixed strings to construct a fully-qualified path.

When an application improperly cleanses special character sequences in user-supplied filenames, a path traversal (or directory traversal) vulnerability may occur. For example, an attacker could specify a filename such as "../../etc/passwd", which resolves to a file outside of the intended directory that the attacker would not normally be authorized to view.

Recommendations

Assume all user-supplied input is malicious. Validate all user-supplied input to ensure that it conforms to the expected format, using centralized data validation routines when possible. When using black lists, be sure that the sanitizing routine performs a sufficient number of iterations to remove all instances of disallowed characters and ensure that the end result is not dangerous.

Associated Flaws by CWE ID:

External Control of File Name or Path (CWE ID 73)(18 flaws)

Fix Required by Policy

Description

This call contains a path manipulation flaw. The argument to the function is a filename constructed using untrusted input. If an attacker is allowed to specify all or part of the filename, it may be possible to gain unauthorized access to files on the server, including those outside the webroot, that would be normally be inaccessible to end users. The level of exposure depends on the effectiveness of input validation routines, if any.

Effort to Fix: 2 - Implementation error. Fix is approx. 6-50 lines of code. 1 day to fix.

Recommendations

Validate all untrusted input to ensure that it conforms to the expected format, using centralized data validation routines when possible. When using black lists, be sure that the sanitizing routine performs a sufficient number of iterations to remove all instances of disallowed characters.

Instances found via Static Scan

	Flaw Id	Module #	Class #	Module	Location	Fix By
	251	2	-	workfluxor.dll	.../capacitycontroller.cs 673	6/21/18
	392	7	-	workfluxor.dll	projects/.../exportexcel.cs 166	6/21/18
	473	7	-	workfluxor.dll	projects/.../exportexcel.cs 270	6/21/18
	275	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 1794	6/21/18
	748	14	-	workfluxor.dll	.../reportscontroller.cs 3471	7/1/18
	731	14	-	workfluxor.dll	.../reportscontroller.cs 3473	7/1/18
	468	16	-	workfluxor.dll	.../requestcontroller.cs 2359	6/21/18
	362	16	-	workfluxor.dll	.../requestcontroller.cs 3234	6/21/18
	192	18	-	workfluxor.dll	projects/.../usercontroller.cs 951	6/21/18
	242	18	-	workfluxor.dll	projects/.../usercontroller.cs 1702	6/21/18
	277	18	-	workfluxor.dll	projects/.../usercontroller.cs 1831	6/21/18
	474	18	-	workfluxor.dll	projects/.../usercontroller.cs 1877	6/21/18
NEW	805	18	-	workfluxor.dll	projects/.../usercontroller.cs 1879	7/3/18
NEW	808	18	-	workfluxor.dll	projects/.../usercontroller.cs 1879	7/3/18
	712	18	-	workfluxor.dll	projects/.../usercontroller.cs 1952	7/1/18

	Flaw Id	Module #	Class #	Module	Location	Fix By
	732	18	-	workfluxor.dll	projects/.../usercontroller.cs 1954	7/1/18
	356	19	-	workfluxor.dll	projects/.../helpers/utility.cs 46	6/21/18
	570	20	-	workfluxor.dll	.../utilitycontroller.cs 72	6/21/18

Insufficient Input Validation(5 flaws)

Description

Weaknesses in this category are related to an absent or incorrect protection mechanism that fails to properly validate input that can affect the control flow or data flow of a program.

Recommendations

Validate input from untrusted sources before it is used. The untrusted data sources may include HTTP requests, file systems, databases, and any external systems that provide data to the application. In the case of HTTP requests, validate all parts of the request, including headers, form fields, cookies, and URL components that are used to transfer information from the browser to the server side application.

Duplicate any client-side checks on the server side. This should be simple to implement in terms of time and difficulty, and will greatly reduce the likelihood of insecure parameter values being used in the application.

Associated Flaws by CWE ID:

Improperly Controlled Modification of Dynamically-Determined Object Attributes (CWE ID 915)(5 flaws)

Description

The software receives input from an upstream component that specifies multiple attributes, properties, or fields that are to be initialized or updated in an object, but it does not properly control which attributes can be modified. By manipulating the contents of an HTTP request, the attacker may be able to set attributes beyond those intended by the developer.

Effort to Fix: 2 - Implementation error. Fix is approx. 6-50 lines of code. 1 day to fix.

Recommendations

In .NET, explicitly specify which of the Model's attributes should be accessible using the Bind attribute by setting the Include property to each allowable property. If it is not feasible to use the preferred property Include, instead specify which attributes should not be accessible using the Bind attribute and setting the Exclude property to each prohibited property.

In Ruby on Rails, explicitly specify which of each model's attributes should be accessible using the attr_accessible method. If for some reason it is not feasible to use the preferred method, specify which attributes should not be accessible in this way using the attr_protected method.

Instances found via Static Scan

Flaw Id	Module #	Class #	Module	Location	Fix By
475	1	-	workfluxor.dll	.../accountcontroller.cs 49	6/21/18
272	16	-	workfluxor.dll	.../requestcontroller.cs 644	6/21/18
752	16	-	workfluxor.dll	.../requestcontroller.cs 1968	7/1/18
505	18	-	workfluxor.dll	projects/.../usercontroller.cs 525	6/21/18
224	18	-	workfluxor.dll	projects/.../usercontroller.cs 801	6/21/18

Low (154 flaws)

Insufficient Input Validation(154 flaws)

Description

Weaknesses in this category are related to an absent or incorrect protection mechanism that fails to properly validate input that can affect the control flow or data flow of a program.

Recommendations

Validate input from untrusted sources before it is used. The untrusted data sources may include HTTP requests, file systems, databases, and any external systems that provide data to the application. In the case of HTTP requests, validate all parts of the request, including headers, form fields, cookies, and URL components that are used to transfer information from the browser to the server side application.

Duplicate any client-side checks on the server side. This should be simple to implement in terms of time and difficulty, and will greatly reduce the likelihood of insecure parameter values being used in the application.

Associated Flaws by CWE ID:

DEPRECATED: Technology-Specific Input Validation Problems (CWE ID 100)(154 flaws)

Description

Weaknesses in this category are caused by inadequately implemented input validation within particular technologies.

Effort to Fix: 3 - Complex implementation error. Fix is approx. 51-500 lines of code. Up to 5 days to fix.

Instances found via Static Scan

	Flaw Id	Module #	Class #	Module	Location	Fix By
NEW	793	1	-	workfluxor.dll	.../accountcontroller.cs 33	
	755	1	-	workfluxor.dll	.../accountcontroller.cs 152	
	718	1	-	workfluxor.dll	.../accountcontroller.cs 216	
NEW	795	2	-	workfluxor.dll	.../capacitycontroller.cs 168	
NEW	801	2	-	workfluxor.dll	.../capacitycontroller.cs 242	
NEW	794	2	-	workfluxor.dll	.../capacitycontroller.cs 535	
	707	3	-	workfluxor.dll	.../models/commonvaluemodel.cs 191	
	762	3	-	workfluxor.dll	.../models/commonvaluemodel.cs 191	

	Flaw Id	Module #	Class #	Module	Location	Fix By
	751	6	-	workfluxor.dll	.../controllers/errorcontroller.cs 24	
	694	8	-	workfluxor.dll	.../feedbackcontroller.cs 61	
	736	8	-	workfluxor.dll	.../feedbackcontroller.cs 618	
NEW	791	8	-	workfluxor.dll	.../feedbackcontroller.cs 644	
	733	8	-	workfluxor.dll	.../feedbackcontroller.cs 807	
	725	8	-	workfluxor.dll	.../feedbackcontroller.cs 893	
	337	9	-	workfluxor.services.dll	.../forwardlookingutilizationentity.cs 38	
NEW	781	10	-	workfluxor.dll	projects/.../homecontroller.cs 209	
	709	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 64	
	467	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 167	
	706	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 212	
	588	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 283	
	749	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 329	
	301	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 400	
	730	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 446	
	336	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 516	
	757	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 562	
	333	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 633	
	729	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 678	
	532	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 749	
	745	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 795	
	563	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 866	
	194	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 911	
	443	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 1018	
	335	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 1096	
	330	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 1253	
	274	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 1298	
	413	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 1402	
	361	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 2049	
	338	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 2232	
	243	11	-	workfluxor.dll	.../miscellaneouscontroller.cs 2270	
NEW	780	12	-	workfluxor.dll	.../reccuranceapicontroller.cs 24	
	562	13	-	workfluxor.services.dll	projects/.../reportdetails.cs 407	
	187	13	-	workfluxor.services.dll	projects/.../reportdetails.cs 407	
	572	13	-	workfluxor.services.dll	projects/.../reportdetails.cs 407	
	446	13	-	workfluxor.services.dll	projects/.../reportdetails.cs 407	
	364	13	-	workfluxor.services.	projects/.../reportdetails.cs 407	

	Flaw Id	Module #	Class #	Module	Location	Fix By
				dll		
NEW	789	14	-	workfluxor.dll	.../reportscontroller.cs 232	
NEW	797	14	-	workfluxor.dll	.../reportscontroller.cs 395	
	705	14	-	workfluxor.dll	.../reportscontroller.cs 963	
	363	14	-	workfluxor.dll	.../reportscontroller.cs 1598	
	753	14	-	workfluxor.dll	.../reportscontroller.cs 1839	
	417	14	-	workfluxor.dll	.../reportscontroller.cs 2095	
	743	14	-	workfluxor.dll	.../reportscontroller.cs 2422	
	747	14	-	workfluxor.dll	.../reportscontroller.cs 3138	
	412	14	-	workfluxor.dll	.../reportscontroller.cs 3192	
	531	14	-	workfluxor.dll	.../reportscontroller.cs 3273	
	722	14	-	workfluxor.dll	.../reportscontroller.cs 3319	
	740	14	-	workfluxor.dll	.../reportscontroller.cs 3464	
	218	14	-	workfluxor.dll	.../reportscontroller.cs 3509	
	708	14	-	workfluxor.dll	.../reportscontroller.cs 3529	
	592	15	-	workfluxor.dll	.../repositorycontroller.cs 74	
	720	16	-	workfluxor.dll	.../requestcontroller.cs 101	
	357	16	-	workfluxor.dll	.../requestcontroller.cs 424	
	535	16	-	workfluxor.dll	.../requestcontroller.cs 644	
NEW	796	16	-	workfluxor.dll	.../requestcontroller.cs 806	
NEW	790	16	-	workfluxor.dll	.../requestcontroller.cs 829	
NEW	802	16	-	workfluxor.dll	.../requestcontroller.cs 900	
NEW	799	16	-	workfluxor.dll	.../requestcontroller.cs 947	
	761	16	-	workfluxor.dll	.../requestcontroller.cs 1357	
	746	16	-	workfluxor.dll	.../requestcontroller.cs 1412	
	695	16	-	workfluxor.dll	.../requestcontroller.cs 1445	
	754	16	-	workfluxor.dll	.../requestcontroller.cs 1478	
NEW	787	16	-	workfluxor.dll	.../requestcontroller.cs 1645	
NEW	798	16	-	workfluxor.dll	.../requestcontroller.cs 1693	
	744	16	-	workfluxor.dll	.../requestcontroller.cs 1758	
	760	16	-	workfluxor.dll	.../requestcontroller.cs 1798	
	741	16	-	workfluxor.dll	.../requestcontroller.cs 1845	
	727	16	-	workfluxor.dll	.../requestcontroller.cs 1884	
	680	16	-	workfluxor.dll	.../requestcontroller.cs 1942	
	739	16	-	workfluxor.dll	.../requestcontroller.cs 1968	
	681	16	-	workfluxor.dll	.../requestcontroller.cs 1977	
	696	16	-	workfluxor.dll	.../requestcontroller.cs 2036	
	724	16	-	workfluxor.dll	.../requestcontroller.cs 2100	
	710	16	-	workfluxor.dll	.../requestcontroller.cs 2171	

	Flaw Id	Module #	Class #	Module	Location	Fix By
	711	16	-	workfluxor.dll	.../requestcontroller.cs 2257	
	726	16	-	workfluxor.dll	.../requestcontroller.cs 2287	
NEW	788	16	-	workfluxor.dll	.../requestcontroller.cs 3230	
	46	-	47	workfluxor.services.dll	string get_ClientRequestID() 0%	
NEW	782	-	47	workfluxor.services.dll	string get_Column1() 0%	
	51	-	1	workfluxor.dll	string get_Column1() 0%	
	177	-	2	workfluxor.dll	string get_Column1() 0%	
NEW	783	-	47	workfluxor.services.dll	string get_Column2() 0%	
	150	-	1	workfluxor.dll	string get_Column2() 0%	
	201	-	2	workfluxor.dll	string get_Column2() 0%	
NEW	784	-	47	workfluxor.services.dll	string get_Column3() 0%	
	53	-	1	workfluxor.dll	string get_Column3() 0%	
	151	-	2	workfluxor.dll	string get_Column3() 0%	
NEW	785	-	47	workfluxor.services.dll	string get_Column4() 0%	
	54	-	1	workfluxor.dll	string get_Column4() 0%	
	152	-	2	workfluxor.dll	string get_Column4() 0%	
NEW	786	-	47	workfluxor.services.dll	string get_Column5() 0%	
	55	-	1	workfluxor.dll	string get_Column5() 0%	
	153	-	2	workfluxor.dll	string get_Column5() 0%	
	234	-	15	workfluxor.dll	string get_Comment() 0%	
	665	-	12	workfluxor.dll	string get_Comment() 0%	
	50	-	47	workfluxor.services.dll	string get_CopyDeliveryTo() 0%	
	676	-	2	workfluxor.dll	string get_CustomID() 0%	
	766	-	16	workfluxor.dll	string get_DeadLineDate() 0%	
	49	-	47	workfluxor.services.dll	string get_DealCode() 0%	
	175	-	2	workfluxor.dll	string get_MaksTeamDeadLineDate() 0%	
	176	-	2	workfluxor.dll	string get_MaksTeamDeadLineTime() 0%	
	61	-	15	workfluxor.dll	string get_ManagerComment() 0%	
	229	-	2	workfluxor.dll	string get_ModifiedBy() 0%	
	232	-	2	workfluxor.dll	string get_ModifiedOn() 0%	
	367	-	1	workfluxor.dll	string get_NoOfDeliverable() 0%	
	424	-	1	workfluxor.dll	string get_Occurrences() 0%	
	283	-	1	workfluxor.dll	string get_OtherProductType() 0%	
	44	-	2	workfluxor.dll	string get_OtherProductType() 0%	

65 Network Drive, Burlington, MA 01803

Tel.+1.339.674.2500 **Fax.**+1.339.674.2502 **URL:**http://www.veracode.com

	Flaw Id	Module #	Class #	Module	Location	Fix By
	102	-	60	workfluxor.services.dll	string get_Password() 0%	
	408	-	31	workfluxor.services.dll	string get_Priority() 0%	
	312	-	1	workfluxor.dll	string get_ProjectCode() 0%	
	284	-	1	workfluxor.dll	string get_ProjectName() 0%	
	690	-	2	workfluxor.dll	string get_ProjectName() 0%	
	423	-	1	workfluxor.dll	string get_RecurrencePattern() 0%	
	584	-	36	workfluxor.services.dll	string get_region() 0%	
	432	-	47	workfluxor.services.dll	string get_RegionName() 0%	
	461	-	47	workfluxor.services.dll	string get_RequestDeadline() 0%	
	48	-	2	workfluxor.dll	string get_RequestDetail() 0%	
	11	-	1	workfluxor.dll	string get_RequestDetails() 0%	
	661	-	16	workfluxor.dll	string get_RequestId() 0%	
	256	-	2	workfluxor.dll	string get_ScopingTime() 0%	
	433	-	47	workfluxor.services.dll	string get_SectorName() 0%	
	258	-	2	workfluxor.dll	string get_Sources() 0%	
	58	-	1	workfluxor.dll	string get_StartDate() 0%	
	379	-	57	workfluxor.services.dll	string get_Status() 0%	
	72	-	15	workfluxor.dll	string get_Status() 0%	
	678	-	60	workfluxor.services.dll	string get_statusCode() 0%	
	253	-	1	workfluxor.dll	string get_SubProductType() 0%	
	260	-	26	workfluxor.services.dll	string get_SurveyDescription() 0%	
	259	-	26	workfluxor.services.dll	string get_SurveyName() 0%	
	677	-	15	workfluxor.dll	string get_Tags() 0%	
	394	-	1	workfluxor.dll	string get_TimePicker() 0%	
	231	-	2	workfluxor.dll	string get_TimeTracking() 0%	
	714	-	20	workfluxor.dll	string get_TrendChart() 0%	
	688	-	3	workfluxor.dll	string get_Type() 0%	
	728	18	-	workfluxor.dll	projects/.../usercontroller.cs 78	
	756	18	-	workfluxor.dll	projects/.../usercontroller.cs 266	
	498	18	-	workfluxor.dll	projects/.../usercontroller.cs 525	
NEW	792	18	-	workfluxor.dll	projects/.../usercontroller.cs 631	
NEW	800	18	-	workfluxor.dll	projects/.../usercontroller.cs 744	
	478	18	-	workfluxor.dll	projects/.../usercontroller.cs 801	
	721	18	-	workfluxor.dll	projects/.../usercontroller.cs 835	

Flaw Id	Module #	Class #	Module	Location	Fix By
719	18	-	workfluxor.dll	projects/.../usercontroller.cs 1862	
737	18	-	workfluxor.dll	projects/.../usercontroller.cs 1904	
742	21	-	workfluxor.dll	.../utilizationforecastcontroller.cs 38	

Very Low (0 flaws)

No flaws of this type were found

Info (0 flaws)

No flaws of this type were found

Flaws in Common Modules

This section highlights the score impact of flaws in common modules in this application.

Module: Workfluxor.Services.dll Used by 4 executables; score impact: 1

Location	Severity	# Instances	Flaw Category	CWE ID	Exploitability
workfluxor_services_dll.Workfluxor.Services.Entity.RequestDetailsById_Get_Data					
get_Column2 0%	2	1	Insufficient Input Validation	100	Neutral
get_Column3 0%	2	1	Insufficient Input Validation	100	Neutral
get_Column4 0%	2	1	Insufficient Input Validation	100	Neutral
get_SectorName 0%	2	1	Insufficient Input Validation	100	Neutral
get_RequestDeadline 0%	2	1	Insufficient Input Validation	100	Neutral
get_Column5 0%	2	1	Insufficient Input Validation	100	Neutral
get_ClientRequestID 0%	2	1	Insufficient Input Validation	100	Neutral
get_DealCode 0%	2	1	Insufficient Input Validation	100	Neutral
get_RegionName 0%	2	1	Insufficient Input Validation	100	Neutral
get_CopyDeliveryTo 0%	2	1	Insufficient Input Validation	100	Neutral
get_Column1 0%	2	1	Insufficient Input Validation	100	Neutral
workfluxor_services_dll.Workfluxor.Services.Entity.FbSurveyWithQuestionAndChoiceEntity					
get_SurveyDescription 0%	2	1	Insufficient Input Validation	100	Neutral
get_SurveyName 0%	2	1	Insufficient Input Validation	100	Neutral
workfluxor_services_dll.Workfluxor.Services.Entity.UserDetails					
get_Password 0%	2	1	Insufficient Input Validation	100	Neutral
get_statusCode 0%	2	1	Insufficient Input Validation	100	Neutral
forwardlookingutilizationentity.cs 38	2	1	Insufficient Input Validation	100	Neutral
reportdetails.cs 407	2	5	Insufficient Input Validation	100	Neutral
workfluxor_services_dll.Workfluxor.Services.Entity.PriorityDetails					
get_Priority 0%	2	1	Insufficient Input Validation	100	Neutral
workfluxor_services_dll.Workfluxor.Services.Entity.RegionSectorMappingDetailEntity					
get_region 0%	2	1	Insufficient Input Validation	100	Neutral
workfluxor_services_dll.Workfluxor.Services.Entity.StatusDetails					
get_Status 0%	2	1	Insufficient Input Validation	100	Neutral

About Veracode's Methodology

The Veracode platform uses static and dynamic analysis (for web applications) to inspect executables and identify security flaws in your applications. Using both static and dynamic analysis helps reduce false negatives and detect a broader range of security flaws. The static binary analysis engine models the binary executable into an intermediate representation, which is then verified for security flaws using a set of automated security scans. Dynamic analysis uses an automated penetration testing technique to detect security flaws at runtime. Once the automated process is complete, a security technician verifies the output to ensure the lowest false positive rates in the industry. The end result is an accurate list of security flaws for the classes of automated scans applied to the application.

Veracode Rating System Using Multiple Analysis Techniques

Higher assurance applications require more comprehensive analysis to accurately score their security quality. Because each analysis technique (automated static, automated dynamic, manual penetration testing or manual review) has differing false negative (FN) rates for different types of security flaws, any single analysis technique or even combination of techniques is bound to produce a certain level of false negatives. Some false negatives are acceptable for lower business critical applications, so a less expensive analysis using only one or two analysis techniques is acceptable. At higher business criticality the FN rate should be close to zero, so multiple analysis techniques are recommended.

Application Security Policies

The Veracode platform allows an organization to define and enforce a uniform application security policy across all applications in its portfolio. The elements of an application security policy include the target Veracode Level for the application; types of flaws that should not be in the application (which may be defined by flaw severity, flaw category, CWE, or a common standard including OWASP, CWE/SANS Top 25, or PCI); minimum Veracode security score; required scan types and frequencies; and grace period within which any policy-relevant flaws should be fixed.

Policy constraints

Policies have three main constraints that can be applied: rules, required scans, and remediation grace periods.

Evaluating applications against a policy

When an application is evaluated against a policy, it can receive one of four assessments:

Not assessed The application has not yet had a scan published
Passed The application has passed all the aspects of the policy, including rules, required scans, and grace period.
Did not pass The application has not completed all required scans; has not achieved the target Veracode Level; or has one or more policy relevant flaws that have exceeded the grace period to fix.
Conditional pass The application has one or more policy relevant flaws that have not yet exceeded the grace period to fix.

Understand Veracode Levels

The Veracode Level (VL) achieved by an application is determined by type of testing performed on the application, and the severity and types of flaws detected. A minimum security score (defined below) is also required for each level.

There are five Veracode Levels denoted as VL1, VL2, VL3, VL4, and VL5. VL1 is the lowest level and is achieved by demonstrating that security testing, automated static or dynamic, is utilized during the SDLC. VL5 is the highest level and is achieved by performing automated and manual testing and removing all significant flaws. The Veracode Levels VL2, VL3, and VL4 form a continuum of increasing software assurance between VL1 and VL5.

For IT staff operating applications, Veracode Levels can be used to set application security policies. For deployment scenarios of different business criticality, differing VLs should be made requirements. For example, the policy for applications that handle credit card transactions, and therefore have PCI compliance requirements, should be VL5. A medium business criticality internal application could have a policy requiring VL3.

Software developers can decide which VL they want to achieve based on the requirements of their customers. Developers of software that is mission critical to most of their customers will want to achieve VL5. Developers of general purpose business software may want

to achieve VL3 or VL4. Once the software has achieved a Veracode Level it can be communicated to customers through a Veracode Report or through the Veracode Directory on the Veracode web site.

Criteria for achieving Veracode Levels

The following table defines the details to achieve each Veracode Level. The criteria for all columns: Flaw Severities Not Allowed, Flaw Categories not Allowed, Testing Required, and Minimum Score.

*Dynamic is only an option for web applications.

Veracode Level	Flaw Severities Not Allowed	Testing Required*	Minimum Score
VL5	V.High, High, Medium	Static AND Manual	90
VL4	V.High, High, Medium	Static	80
VL3	V.High, High	Static	70
VL2	V.High	Static OR Dynamic OR Manual	60
VL1		Static OR Dynamic OR Manual	

When multiple testing techniques are used it is likely that not all testing will be performed on the exact same build. If that is the case the latest test results from a particular technique will be used to calculate the current Veracode Level. After 6 months test results will be deemed out of date and will no longer be used to calculate the current Veracode Level.

Business Criticality

The foundation of the Veracode rating system is the concept that more critical applications require higher security quality scores to be acceptable risks. Less business critical applications can tolerate lower security quality. The business criticality is dictated by the typical deployed environment and the value of data used by the application. Factors that determine business criticality are: reputation damage, financial loss, operational risk, sensitive information disclosure, personal safety, and legal violations.

US. Govt. OMB Memorandum M-04-04; NIST FIPS Pub. 199

Business Criticality Description

Very High	Mission critical for business/safety of life and limb on the line
High	Exploitation causes serious brand damage and financial loss with long term business impact
Medium	Applications connected to the internet that process financial or private customer information
Low	Typically internal applications with non-critical business impact
Very Low	Applications with no material business impact

Business Criticality Definitions

Very High (BC5) This is typically an application where the safety of life or limb is dependent on the system; it is mission critical the application maintain 100% availability for the long term viability of the project or business. Examples are control software for industrial, transportation or medical equipment or critical business systems such as financial trading systems.
High (BC4) This is typically an important multi-user business application reachable from the internet and is critical that the application maintain high availability to accomplish its mission. Exploitation of high criticality applications cause serious brand damage and business/financial loss and could lead to long term business impact.
Medium (BC3) This is typically a multi-user application connected to the internet or any system that processes financial or private customer information. Exploitation of medium criticality applications typically result in material business impact resulting

in some financial loss, brand damage or business liability. An example is a financial services company's internal 401K management system.

Low (BC2) This is typically an internal only application that requires low levels of application security such as authentication to protect access to non-critical business information and prevent IT disruptions. Exploitation of low criticality applications may lead to minor levels of inconvenience, distress or IT disruption. An example internal system is a conference room reservation or business card order system.

Very Low (BC1) Applications that have no material business impact should its confidentiality, data integrity and availability be affected. Code security analysis is not required for applications at this business criticality, and security spending should be directed to other higher criticality applications.

Scoring Methodology

The Veracode scoring system, Security Quality Score, is built on the foundation of two industry standards, the Common Weakness Enumeration (CWE) and Common Vulnerability Scoring System (CVSS). CWE provides the dictionary of security flaws and CVSS provides the foundation for computing severity, based on the potential Confidentiality, Integrity and Availability impact of a flaw if exploited.

The Security Quality Score is a single score from 0 to 100, where 0 is the most insecure application and 100 is an application with no detectable security flaws. The score calculation includes non-linear factors so that, for instance, a single Severity 5 flaw is weighted more heavily than five Severity 1 flaws, and so that each additional flaw at a given severity contributes progressively less to the score.

Veracode assigns a severity level to each flaw type based on three foundational application security requirements — Confidentiality, Integrity and Availability. Each of the severity levels reflects the potential business impact if a security breach occurs across one or more of these security dimensions.

Confidentiality Impact

According to CVSS, this metric measures the impact on confidentiality if a exploit should occur using the vulnerability on the target system. At the weakness level, the scope of the Confidentiality in this model is within an application and is measured at three levels of impact -None, Partial and Complete.

Integrity Impact

This metric measures the potential impact on integrity of the application being analyzed. Integrity refers to the trustworthiness and guaranteed veracity of information within the application. Integrity measures are meant to protect data from unauthorized modification. When the integrity of a system is sound, it is fully proof from unauthorized modification of its contents.

Availability Impact

This metric measures the potential impact on availability if a successful exploit of the vulnerability is carried out on a target application. Availability refers to the accessibility of information resources. Almost exclusive to this domain are denial-of-service vulnerabilities. Attacks that compromise authentication and authorization for application access, application memory, and administrative privileges are examples of impact on the availability of an application.

Security Quality Score Calculation

The overall Security Quality Score is computed by aggregating impact levels of all weaknesses within an application and representing the score on a 100 point scale. This score does not predict vulnerability potential as much as it enumerates the security weaknesses and their impact levels within the application code.

The Raw Score formula puts weights on each flaw based on its impact level. These weights are exponential and determined by empirical analysis by Veracode's application security experts with validation from industry experts. The score is normalized to a scale of 0 to 100, where a score of 100 is an application with 0 detected flaws using the analysis technique for the application's business criticality.

Understand Severity, Exploitability, and Remediation Effort

Severity and exploitability are two different measures of the seriousness of a flaw. Severity is defined in terms of the potential impact to confidentiality, integrity, and availability of the application as defined in the CVSS, and exploitability is defined in terms of the likelihood

or ease with which a flaw can be exploited. A high severity flaw with a high likelihood of being exploited by an attacker is potentially more dangerous than a high severity flaw with a low likelihood of being exploited.

Remediation effort, also called Complexity of Fix, is a measure of the likely effort required to fix a flaw. Together with severity, the remediation effort is used to give Fix First guidance to the developer.

Veracode Flaw Severities

Veracode flaw severities are defined as follows:

Severity	Description
Very High	The offending line or lines of code is a very serious weakness and is an easy target for an attacker. The code should be modified immediately to avoid potential attacks.
High	The offending line or lines of code have significant weakness, and the code should be modified immediately to avoid potential attacks.
Medium	A weakness of average severity. These should be fixed in high assurance software. A fix for this weakness should be considered after fixing the very high and high for medium assurance software.
Low	This is a low priority weakness that will have a small impact on the security of the software. Fixing should be consideration for high assurance software. Medium and low assurance software can ignore these flaws.
Very Low	Minor problems that some high assurance software may want to be aware of. These flaws can be safely ignored in medium and low assurance software.
Informational	Issues that have no impact on the security quality of the application but which may be of interest to the reviewer.

Informational findings

Informational severity findings are items observed in the analysis of the application that have no impact on the security quality of the application but may be interesting to the reviewer for other reasons. These findings may include code quality issues, API usage, and other factors.

Informational severity findings have no impact on the security quality score of the application and are not included in the summary tables of flaws for the application.

Exploitability

Each flaw instance in a static scan may receive an exploitability rating. The rating is an indication of the intrinsic likelihood that the flaw may be exploited by an attacker. Veracode recommends that the exploitability rating be used to prioritize flaw remediation within a particular group of flaws with the same severity and difficulty of fix classification.

The possible exploitability ratings include:

Exploitability	Description
V. Unlikely	Very unlikely to be exploited
Unlikely	Unlikely to be exploited

Exploitability	Description
Neutral	Neither likely nor unlikely to be exploited.
Likely	Likely to be exploited
V. Likely	Very likely to be exploited

Note: All reported flaws found via dynamic scans are assumed to be exploitable, because the dynamic scan actually executes the attack in question and verifies that it is valid.

Effort/Complexity of Fix

Each flaw instance receives an effort/complexity of fix rating based on the classification of the flaw. The effort/complexity of fix rating is given on a scale of 1 to 5, as follows:

Effort/Complexity of Fix	Description
5	Complex design error. Requires significant redesign.
4	Simple design error. Requires redesign and up to 5 days to fix.
3	Complex implementation error. Fix is approx. 51-500 lines of code. Up to 5 days to fix.
2	Implementation error. Fix is approx. 6-50 lines of code. 1 day to fix.
1	Trivial implementation error. Fix is up to 5 lines of code. One hour or less to fix.

Flaw Types by Severity Level

The flaw types by severity level table provides a summary of flaws found in the application by Severity and Category. The table puts the Security Quality Score into context by showing the specific breakout of flaws by severity, used to compute the score as described above. If multiple analysis techniques are used, the table includes a breakout of all flaws by category and severity for each analysis type performed.

Flaws by Severity

The flaws by severity chart shows the distribution of flaws by severity. An application can get a mediocre security rating by having a few high risk flaws or many medium risk flaws.

Flaws in Common Modules

The flaws in common modules listing shows a summary of flaws in shared dependency modules in this application. A shared dependency is a dependency that is used by more than one analyzed module. Each module is listed with the number of executables that consume it as a dependency and a summary of the impact on the application's security score of the flaws found in the dependency.

The score impact represents the amount that the application score would increase if all the flaws in the shared dependency module were fixed. This information can be used to focus remediation efforts on common modules with a higher impact on the application security score.

Only common modules that were uploaded with debug information are included in the Flaws in Common Modules listing.

Action Items

The Action Items section of the report provides guidance on the steps required to bring the application to a state where it passes its assigned policy. These steps may include fixing or mitigating flaws or performing additional scans. The section also includes best practice recommendations to improve the security quality of the application.

Common Weakness Enumeration (CWE)

The Common Weakness Enumeration (CWE) is an industry standard classification of types of software weaknesses, or flaws, that can lead to security problems. CWE is widely used to provide a standard taxonomy of software errors. Every flaw in a Veracode report is classified according to a standard CWE identifier.

More guidance and background about the CWE is available at http://cwe.mitre.org/data/index.html.

About Manual Assessments

The Veracode platform can include the results from a manual assessment (usually a penetration test or code review) as part of a report. These results differ from the results of automated scans in several important ways, including objectives, attack vectors, and common attack patterns.

A manual penetration assessment is conducted to observe the application code in a run-time environment and to simulate real-world attack scenarios. Manual testing is able to identify design flaws, evaluate environmental conditions, compound multiple lower risk flaws into higher risk vulnerabilities, and determine if identified flaws affect the confidentiality, integrity, or availability of the application.

Objectives

The stated objectives of a manual penetration assessment are:

- Perform testing, using proprietary and/or public tools, to determine whether it is possible for an attacker to:
- Circumvent authentication and authorization mechanisms
- Escalate application user privileges
- Hijack accounts belonging to other users
- Violate access controls placed by the site administrator
- Alter data or data presentation
- Corrupt application and data integrity, functionality and performance
- Circumvent application business logic
- Circumvent application session management
- Break or analyze use of cryptography within user accessible components
- Determine possible extent access or impact to the system by attempting to exploit vulnerabilities
- Score vulnerabilities using the Common Vulnerability Scoring System (CVSS)
- Provide tactical recommendations to address security issues of immediate consequence

Provide strategic recommendations to enhance security by leveraging industry best practices

Attack vectors

In order to achieve the stated objectives, the following tests are performed as part of the manual penetration assessment, when applicable to the platforms and technologies in use:

- Cross Site Scripting (XSS)
- SQL Injection
- Command Injection
- Cross Site Request Forgery (CSRF)
- Authentication/Authorization Bypass
- Session Management testing, e.g. token analysis, session expiration, and logout effectiveness
- Account Management testing, e.g. password strength, password reset, account lockout, etc.
- Directory Traversal
- Response Splitting
- Stack/Heap Overflows
- Format String Attacks

- Cookie Analysis
- Server Side Includes Injection
- Remote File Inclusion
- LDAP Injection
- XPATH Injection
- Internationalization attacks
- Denial of Service testing at the application layer only
- AJAX Endpoint Analysis
- Web Services Endpoint Analysis
- HTTP Method Analysis
- SSL Certificate and Cipher Strength Analysis
- Forced Browsing

CAPEC Attack Pattern Classification

The following attack pattern classifications are used to group similar application flaws discovered during manual penetration testing. Attack patterns describe the general methods employed to access and exploit the specific weaknesses that exist within an application. CAPEC (Common Attack Pattern Enumeration and Classification) is an effort led by Cigital, Inc. and is sponsored by the United States Department of Homeland Security's National Cyber Security Division.

Abuse of Functionality

Exploitation of business logic errors or misappropriation of programmatic resources. Application functions are developed to specifications with particular intentions, and these types of attacks serve to undermine those intentions.

Examples:

- Exploiting password recovery mechanisms
- Accessing unpublished or test APIs
- Cache poisoning

Spoofing

Impersonation of entities or trusted resources. A successful attack will present itself to a verifying entity with an acceptable level of authenticity.

Examples:

- Man in the middle attacks
- Checksum spoofing
- Phishing attacks

Probabilistic Techniques

Using predictive capabilities or exhaustive search techniques in order to derive or manipulate sensitive information. Attacks capitalize on the availability of computing resources or the lack of entropy within targeted components.

Examples:

- Password brute forcing
- Cryptanalysis
- Manipulation of authentication tokens

Exploitation of Authentication

Circumventing authentication requirements to access protected resources. Design or implementation flaws may allow authentication checks to be ignored, delegated, or bypassed.

Examples:

- Cross-site request forgery
- Reuse of session identifiers
- Flawed authentication protocol

Resource Depletion

Affecting the availability of application components or resources through symmetric or asymmetric consumption. Unrestricted access to computationally expensive functions or implementation flaws that affect the stability of the application can be targeted by an attacker in order to cause denial of service conditions.

Examples:

- Flooding attacks
- Unlimited file upload size
- Memory leaks

Exploitation of Privilege/Trust

Undermining the application's trust model in order to gain access to protected resources or gain additional levels of access as defined by the application. Applications that implicitly extend trust to resources or entities outside of their direct control are susceptible to attack.

Examples:

- Insufficient access control lists
- Circumvention of client side protections
- Manipulation of role identification information

Injection

Inserting unexpected inputs to manipulate control flow or alter normal business processing. Applications must contain sufficient data validation checks in order to sanitize tainted data and prevent malicious, external control over internal processing.

Examples:

- SQL Injection
- Cross-site scripting
- XML Injection

Data Structure Attacks

Supplying unexpected or excessive data that results in more data being written to a buffer than it is capable of holding. Successful attacks of this class can result in arbitrary command execution or denial of service conditions.

Examples:

- Buffer overflow
- Integer overflow
- Format string overflow

Data Leakage Attacks

Recovering information exposed by the application that may itself be confidential or may be useful to an attacker in discovering or exploiting other weaknesses. A successful attack may be conducted passive observation or active interception methods. This attack pattern often manifests itself in the form of applications that expose sensitive information within error messages.

Examples:

- Sniffing clear-text communication protocols
- Stack traces returned to end users
- Sensitive information in HTML comments

Resource Manipulation

Manipulating application dependencies or accessed resources in order to undermine security controls and gain unauthorized access to protected resources. Applications may use tainted data when constructing paths to local resources or when constructing processing environments.

Examples:

- Carriage Return Line Feed log file injection
- File retrieval via path manipulation
- User specification of configuration files

Time and State Attacks

Undermining state condition assumptions made by the application or capitalizing on time delays between security checks and performed operations. An application that does not enforce a required processing sequence or does not handle concurrency adequately will be susceptible to these attack patterns.

Examples:

- Bypassing intermediate form processing steps
- Time-of-check and time-of-use race conditions
- Deadlock triggering to cause a denial of service

Terms of Use

Use and distribution of this report are governed by the agreement between Veracode and its customer. In particular, this report and the results in the report cannot be used publicly in connection with Veracode's name without written permission.

Appendix A: Changes from Last Scan

Latest Scan		Prior Scan	
Static Scan			
Scan Name:	4 Apr 2018 Static	Scan Name:	2 Apr 2018 Static
Completed:	4/4/18	Completed:	4/2/18
Score:	77	Score:	73

Changes in scope of scan (static)

New Modules

Module Name	Compiler	Operating Environment	Engine Version
App_Web__attachmentdownloaddetails.cshtml.50f944d4.dll	MSIL_MSVC11_X86	Win32	120896

Removed modules

Module Name	Compiler	Operating Environment	Engine Version
App_global.asax.dll	MSIL_MSVC11_X86	Win32	120896

Flaws not detected in current scan

The following is a list of all flaws found in the prior scan of this application that were not detected in the current scan.

High (1 flaw)

Fix Required by Policy

Credentials Management(1 flaw)

Associated Flaws by CWE ID:

Use of Hard-coded Password (CWE ID 259)(1 flaw)

Instances found via Static Scan

Flaw Id	Module #	Class #	Module	Location	Fix By
738	5	-	workfluxor.dll	projects/.../helpers/encryption.cs 1	

Medium (2 flaws)

Fix Required by Policy

Directory Traversal(2 flaws)

Associated Flaws by CWE ID:

External Control of File Name or Path (CWE ID 73)(2 flaws)

Fix Required by Policy

Instances found via Static Scan

Flaw Id	Module #	Class #	Module	Location	Fix By
767	18	-	workfluxor.dll	projects/.../usercontroller.cs 1885	

Flaw Id	Module #	Class #	Module	Location	Fix By
768	18	-	workfluxor.dll	projects/.../usercontroller.cs 1885	

Low (294 flaws)

Code Quality(1 flaw)

Associated Flaws by CWE ID:

Improper Resource Shutdown or Release (CWE ID 404)(1 flaw)

Instances found via Static Scan

Flaw Id	Module #	Class #	Module	Location	Fix By
750	4	-	workfluxor.docsearchsearvices.dll	projects/.../docsearch.cs 124	

Insufficient Input Validation(293 flaws)

Associated Flaws by CWE ID:

DEPRECATED: Technology-Specific Input Validation Problems (CWE ID 100)(293 flaws)

Instances found via Static Scan

Flaw Id	Module #	Class #	Module	Location	Fix By
108	9	-	workfluxor.services.dll	.../forwardlookingutilizationentity.cs 32	
117	9	-	workfluxor.services.dll	.../forwardlookingutilizationentity.cs 32	
244	13	-	workfluxor.services.dll	projects/.../reportdetails.cs 328	
442	13	-	workfluxor.services.dll	projects/.../reportdetails.cs 328	
495	13	-	workfluxor.services.dll	projects/.../reportdetails.cs 328	
593	13	-	workfluxor.services.dll	projects/.../reportdetails.cs 328	
217	13	-	workfluxor.services.dll	projects/.../reportdetails.cs 328	
416	13	-	workfluxor.services.dll	projects/.../reportdetails.cs 328	
445	13	-	workfluxor.services.dll	projects/.../reportdetails.cs 328	
360	13	-	workfluxor.services.dll	projects/.../reportdetails.cs 328	
510	13	-	workfluxor.services.dll	projects/.../reportdetails.cs 328	

Flaw Id	Module #	Class #	Module	Location	Fix By
70	-	21	workfluxor.services.dll	string get_AliasFileName() 0%	
262	-	45	workfluxor.services.dll	string get_AliasFileName() 0%	
579	-	53	workfluxor.services.dll	string get_Assign_Date() 0%	
149	-	53	workfluxor.services.dll	string get_AssignedAnalysts() 0%	
576	-	2	workfluxor.dll	string get_AssignTo() 0%	
673	-	15	workfluxor.dll	string get_AttachmentId() 0%	
479	-	1	workfluxor.dll	string get_attachments() 0%	
230	-	2	workfluxor.dll	string get_Attachments() 0%	
366	-	1	workfluxor.dll	string get_AverageEffort() 0%	
252	-	1	workfluxor.dll	string get_BankerID() 0%	
713	-	5	workfluxor.dll	string get_CancellationReason() 0%	
550	-	53	workfluxor.services.dll	string get_Client_Email() 0%	
518	-	53	workfluxor.services.dll	string get_Client_Name() 0%	
340	-	1	workfluxor.dll	string get_ClientENOH() 0%	
581	-	53	workfluxor.services.dll	string get_Closed_Date() 0%	
522	-	46	workfluxor.services.dll	string get_Closed_Date() 0%	
322	-	29	workfluxor.services.dll	string get_Code() 0%	
653	-	12	workfluxor.dll	string get_Comment() 0%	
136	-	21	workfluxor.services.dll	string get_Comment() 0%	
40	-	1	workfluxor.dll	string get_Comments() 0%	
368	-	1	workfluxor.dll	string get_CopyDeliveryTo() 0%	
657	-	24	workfluxor.services.dll	string get_CostCenter() 0%	
239	-	61	workfluxor.services.dll	string get_CostCenter() 0%	
349	-	62	workfluxor.services.dll	string get_CostCenter() 0%	
100	-	60	workfluxor.services.dll	string get_CostCenter() 0%	
380	-	22	workfluxor.services.dll	string get_CountryName() 0%	
668	-	62	workfluxor.services.dll	string get_CountryName() 0%	
146	-	53	workfluxor.services.dll	string get_CountryName() 0%	
454	-	23	workfluxor.services.dll	string get_CreatedBy() 0%	

Flaw Id	Module #	Class #	Module	Location	Fix By
209	-	45	workfluxor.services.dll	string get_CreatedOn() 0%	
208	-	53	workfluxor.services.dll	string get_CreatedOn() 0%	
519	-	46	workfluxor.services.dll	string get_CreatedOn() 0%	
45	-	2	workfluxor.dll	string get_CustomID() 0%	
457	-	53	workfluxor.services.dll	string get_CustomID() 0%	
375	-	46	workfluxor.services.dll	string get_CustomID() 0%	
758	-	3	workfluxor.dll	string get_Date() 0%	
655	-	12	workfluxor.dll	string get_Deadline() 0%	
393	-	1	workfluxor.dll	string get_Deadline() 0%	
143	-	2	workfluxor.dll	string get_DeadlineDate() 0%	
553	-	53	workfluxor.services.dll	string get_DeadlineDateTime() 0%	
427	-	46	workfluxor.services.dll	string get_DeadlineDateTime() 0%	
156	-	45	workfluxor.services.dll	string get_DeadlineDateTime() 0%	
651	-	16	workfluxor.dll	string get_DeadLineTime() 0%	
658	-	12	workfluxor.dll	string get_DeadlineTime() 0%	
144	-	2	workfluxor.dll	string get_DeadlineTime() 0%	
138	-	56	workfluxor.services.dll	string get_Description() 0%	
422	-	23	workfluxor.services.dll	string get_DesignationName() 0%	
551	-	53	workfluxor.services.dll	string get_DesignationName() 0%	
68	-	21	workfluxor.services.dll	string get_DisCreatedOn() 0%	
207	-	53	workfluxor.services.dll	string get_DueByMonth() 0%	
488	-	46	workfluxor.services.dll	string get_DueByMonth() 0%	
675	-	46	workfluxor.services.dll	string get_DueByWeek() 0%	
487	-	53	workfluxor.services.dll	string get_DueByWeek() 0%	
483	-	53	workfluxor.services.dll	string get_eBanker_Code() 0%	
578	-	53	workfluxor.services.dll	string get_Efforts_Breakup() 0%	
526	-	33	workfluxor.services.dll	string get_EmailDL() 0%	
101	-	24	workfluxor.services.dll	string get_EmailID() 0%	

Flaw Id	Module #	Class #	Module	Location	Fix By
240	-	61	workfluxor.services.dll	string get_EmailID() 0%	
716	-	62	workfluxor.services.dll	string get_EmailID() 0%	
298	-	60	workfluxor.services.dll	string get_EmailID() 0%	
557	-	45	workfluxor.services.dll	string get_EncryptedId() 0%	
521	-	46	workfluxor.services.dll	string get_EncryptedId() 0%	
546	-	2	workfluxor.dll	string get_EncryptRequestID() 0%	
482	-	1	workfluxor.dll	string get_EndDate() 0%	
511	-	1	workfluxor.dll	string get_EndTime() 0%	
73	-	2	workfluxor.dll	string get_EngagementDescription() 0%	
763	-	8	workfluxor.dll	string get_ErrorResponse() 0%	
682	-	8	workfluxor.dll	string get_ErrorText() 0%	
257	-	2	workfluxor.dll	string get_EstimatedHours() 0%	
429	-	46	workfluxor.services.dll	string get_FBStatus() 0%	
178	-	53	workfluxor.services.dll	string get_FBStatus() 0%	
458	-	46	workfluxor.services.dll	string get_FBSurveryLink() 0%	
35	-	27	workfluxor.services.dll	string get_FeedbackResponse() 0%	
212	-	45	workfluxor.services.dll	string get_FileExtention() 0%	
64	-	21	workfluxor.services.dll	string get_FileName() 0%	
211	-	45	workfluxor.services.dll	string get_FileName() 0%	
62	-	21	workfluxor.services.dll	string get_FilePath() 0%	
236	-	45	workfluxor.services.dll	string get_FilePath() 0%	
523	-	21	workfluxor.services.dll	string get_FileSize() 0%	
490	-	21	workfluxor.services.dll	string get_FileType() 0%	
97	-	24	workfluxor.services.dll	string get_FirstName() 0%	
216	-	61	workfluxor.services.dll	string get_FirstName() 0%	
734	-	60	workfluxor.services.dll	string get_FirstName() 0%	
325	-	62	workfluxor.services.dll	string get_FirstName() 0%	
660	-	17	workfluxor.dll	string get_FolderPath() 0%	

Flaw Id	Module #	Class #	Module	Location	Fix By
650	-	13	workfluxor.dll	string get_Guid() 0%	
671	-	2	workfluxor.dll	string get_Guid() 0%	
685	-	19	workfluxor.dll	string get_Guid() 0%	
697	-	18	workfluxor.dll	string get_Guid() 0%	
715	-	20	workfluxor.dll	string get_Guid() 0%	
723	-	11	workfluxor.dll	string get_Guid() 0%	
547	-	1	workfluxor.dll	string get_Guid() 0%	
686	-	18	workfluxor.dll	string get_ImageData() 0%	
689	-	9	workfluxor.dll	string get_ImagePath() 0%	
667	-	24	workfluxor.services.dll	string get_ImageUrl() 0%	
679	-	60	workfluxor.services.dll	string get_ImageUrl() 0%	
378	-	62	workfluxor.services.dll	string get_ImageUrl() 0%	
99	-	61	workfluxor.services.dll	string get_ImageUrl() 0%	
455	-	1	workfluxor.dll	string get_IncludeWeekEnds() 0%	
515	-	53	workfluxor.services.dll	string get_Industry() 0%	
700	-	4	workfluxor.dll	string get_IsAccepted() 0%	
327	-	62	workfluxor.services.dll	string get_IsActive() 0%	
350	-	62	workfluxor.services.dll	string get_IsPool() 0%	
395	-	1	workfluxor.dll	string get_IsRecurrence() 0%	
292	-	25	workfluxor.services.dll	string get_L1Desktop() 0%	
317	-	25	workfluxor.services.dll	string get_L1Mobile() 0%	
318	-	25	workfluxor.services.dll	string get_L2Desktop() 0%	
319	-	25	workfluxor.services.dll	string get_L2Mobile() 0%	
320	-	25	workfluxor.services.dll	string get_L3Desktop() 0%	
344	-	25	workfluxor.services.dll	string get_L3Mobile() 0%	
345	-	25	workfluxor.services.dll	string get_L4Desktop() 0%	
346	-	25	workfluxor.services.dll	string get_L4Mobile() 0%	
347	-	25	workfluxor.services.dll	string get_L5Desktop() 0%	
374	-	25	workfluxor.services.dll	string get_L5Mobile() 0%	
372	-	41	workfluxor.services.	string get_label() 0%	

Flaw Id	Module #	Class #	Module	Location	Fix By
			dll		
373	-	43	workfluxor.services.dll	string get_label() 0%	
396	-	40	workfluxor.services.dll	string get_label() 0%	
397	-	39	workfluxor.services.dll	string get_label() 0%	
398	-	42	workfluxor.services.dll	string get_label() 0%	
425	-	44	workfluxor.services.dll	string get_label() 0%	
369	-	1	workfluxor.dll	string get_Language() 0%	
321	-	29	workfluxor.services.dll	string get_Language() 0%	
98	-	24	workfluxor.services.dll	string get_LastName() 0%	
238	-	61	workfluxor.services.dll	string get_LastName() 0%	
735	-	60	workfluxor.services.dll	string get_LastName() 0%	
326	-	62	workfluxor.services.dll	string get_LastName() 0%	
382	-	62	workfluxor.services.dll	string get_Manager() 0%	
489	-	21	workfluxor.services.dll	string get_ManagerComment() 0%	
371	-	38	workfluxor.services.dll	string get_Message() 0%	
528	-	33	workfluxor.services.dll	string get_Message() 0%	
404	-	22	workfluxor.services.dll	string get_Message() 0%	
435	-	30	workfluxor.services.dll	string get_Message() 0%	
560	-	34	workfluxor.services.dll	string get_Message() 0%	
39	-	37	workfluxor.services.dll	string get_Message() 0%	
481	-	23	workfluxor.services.dll	string get_Message() 0%	
158	-	36	workfluxor.services.dll	string get_Message() 0%	
463	-	32	workfluxor.services.dll	string get_Message() 0%	
439	-	31	workfluxor.services.dll	string get_Message() 0%	
185	-	58	workfluxor.services.dll	string get_Message() 0%	
297	-	63	workfluxor.services.dll	string get_Message() 0%	

Flaw Id	Module #	Class #	Module	Location	Fix By
717	-	54	workfluxor.services.dll	string get_Message() 0%	
296	-	55	workfluxor.services.dll	string get_Message() 0%	
493	-	56	workfluxor.services.dll	string get_Message() 0%	
215	-	59	workfluxor.services.dll	string get_Message() 0%	
464	-	57	workfluxor.services.dll	string get_Message() 0%	
83	-	46	workfluxor.services.dll	string get_Methodology() 0%	
313	-	1	workfluxor.dll	string get_Methodology() 0%	
405	-	30	workfluxor.services.dll	string get_Methodology() 0%	
456	-	23	workfluxor.services.dll	string get_ModifiedBy() 0%	
656	-	53	workfluxor.services.dll	string get_ModifiedOn() 0%	
210	-	45	workfluxor.services.dll	string get_ModifiedOn() 0%	
228	-	2	workfluxor.dll	string get_ModifiedOn() 0%	
5	-	18	workfluxor.dll	string get_Name() 0%	
764	-	6	workfluxor.dll	string get_NewPassword() 0%	
684	-	6	workfluxor.dll	string get_OldPassword() 0%	
104	-	60	workfluxor.services.dll	string get_OtherDesignation() 0%	
268	-	61	workfluxor.services.dll	string get_OtherDesignation() 0%	
241	-	61	workfluxor.services.dll	string get_Password() 0%	
662	-	60	workfluxor.services.dll	string get_PhoneNumber() 0%	
666	-	24	workfluxor.services.dll	string get_PhoneNumber() 0%	
103	-	61	workfluxor.services.dll	string get_PhoneNumber() 0%	
377	-	62	workfluxor.services.dll	string get_PhoneNumber() 0%	
66	-	21	workfluxor.services.dll	string get_PostedBy() 0%	
580	-	53	workfluxor.services.dll	string get_Processed_Date() 0%	
183	-	45	workfluxor.services.dll	string get_ProductType() 0%	
459	-	46	workfluxor.services.dll	string get_ProductType() 0%	
692	-	32	workfluxor.services.dll	string get_ProductType() 0%	

Flaw Id	Module #	Class #	Module	Location	Fix By
315	-	49	workfluxor.services.dll	string get_ProductType() 0%	
702	-	58	workfluxor.services.dll	string get_ProductType() 0%	
160	-	53	workfluxor.services.dll	string get_ProductType() 0%	
294	-	55	workfluxor.services.dll	string get_ProductType() 0%	
86	-	1	workfluxor.dll	string get_ProductType() 0%	
351	-	62	workfluxor.services.dll	string get_Profile() 0%	
233	-	53	workfluxor.services.dll	string get_ProjectCodeDescription() 0%	
401	-	46	workfluxor.services.dll	string get_ProjectName() 0%	
148	-	53	workfluxor.services.dll	string get_ProjectName() 0%	
47	-	2	workfluxor.dll	string get_ProjectName() 0%	
281	-	1	workfluxor.dll	string get_ProjectType() 0%	
181	-	45	workfluxor.services.dll	string get_ProjectTypeName() 0%	
128	-	46	workfluxor.services.dll	string get_ProjectTypeName() 0%	
436	-	32	workfluxor.services.dll	string get_ProjectTypeName() 0%	
703	-	58	workfluxor.services.dll	string get_ProjectTypeName() 0%	
341	-	50	workfluxor.services.dll	string get_ProjectTypeName() 0%	
161	-	53	workfluxor.services.dll	string get_ProjectTypeName() 0%	
90	-	33	workfluxor.services.dll	string get_ProjectTypeName() 0%	
291	-	25	workfluxor.services.dll	string get_QuestionDescription() 0%	
290	-	25	workfluxor.services.dll	string get_QuestionText() 0%	
512	-	28	workfluxor.services.dll	string get_Region() 0%	
402	-	46	workfluxor.services.dll	string get_Region() 0%	
558	-	34	workfluxor.services.dll	string get_Region() 0%	
285	-	51	workfluxor.services.dll	string get_Region() 0%	
353	-	45	workfluxor.services.dll	string get_Region() 0%	
691	-	55	workfluxor.services.dll	string get_Region() 0%	
41	-	35	workfluxor.services.	string get_Region() 0%	

Flaw Id	Module #	Class #	Module	Location	Fix By
			dll		
147	-	53	workfluxor.services.dll	string get_Region() 0%	
265	-	62	workfluxor.services.dll	string get_Region() 0%	
765	-	6	workfluxor.dll	string get_ReNewPassword() 0%	
399	-	42	workfluxor.services.dll	string get_RequestDate() 0%	
6	-	44	workfluxor.services.dll	string get_RequestDate() 0%	
552	-	53	workfluxor.services.dll	string get_RequestDateTime() 0%	
426	-	46	workfluxor.services.dll	string get_RequestDateTime() 0%	
155	-	45	workfluxor.services.dll	string get_RequestDateTime() 0%	
485	-	53	workfluxor.services.dll	string get_RequestDetail() 0%	
652	-	5	workfluxor.dll	string get_RequestID() 0%	
654	-	12	workfluxor.dll	string get_RequestID() 0%	
659	-	7	workfluxor.dll	string get_RequestId() 0%	
687	-	16	workfluxor.dll	string get_RequestId() 0%	
699	-	4	workfluxor.dll	string get_RequestID() 0%	
431	-	46	workfluxor.services.dll	string get_RequestorEmailID() 0%	
583	-	45	workfluxor.services.dll	string get_RequestorEmailID() 0%	
582	-	45	workfluxor.services.dll	string get_RequestorName() 0%	
430	-	46	workfluxor.services.dll	string get_RequestorName() 0%	
663	-	60	workfluxor.services.dll	string get_responseStatus() 0%	
406	-	62	workfluxor.services.dll	string get_Role() 0%	
513	-	28	workfluxor.services.dll	string get_Sector() 0%	
129	-	46	workfluxor.services.dll	string get_Sector() 0%	
381	-	62	workfluxor.services.dll	string get_Sector() 0%	
154	-	45	workfluxor.services.dll	string get_Sector() 0%	
293	-	55	workfluxor.services.dll	string get_Sector() 0%	
105	-	36	workfluxor.services.dll	string get_Sector() 0%	
287	-	52	workfluxor.services.dll	string get_Sector() 0%	

Flaw Id	Module #	Class #	Module	Location	Fix By
440	-	54	workfluxor.services.dll	string get_Sector() 0%	
516	-	53	workfluxor.services.dll	string get_Sector() 0%	
186	-	59	workfluxor.services.dll	string get_Sector() 0%	
460	-	46	workfluxor.services.dll	string get_ServiceType() 0%	
437	-	32	workfluxor.services.dll	string get_ServiceType() 0%	
693	-	58	workfluxor.services.dll	string get_ServiceType() 0%	
484	-	53	workfluxor.services.dll	string get_ServiceType() 0%	
701	-	56	workfluxor.services.dll	string get_ServiceType() 0%	
159	-	33	workfluxor.services.dll	string get_ServiceType() 0%	
180	-	45	workfluxor.services.dll	string get_ServiceType() 0%	
87	-	1	workfluxor.dll	string get_ServiceType() 0%	
683	-	8	workfluxor.dll	string get_Source() 0%	
649	-	13	workfluxor.dll	string get_Status() 0%	
428	-	46	workfluxor.services.dll	string get_Status() 0%	
286	-	51	workfluxor.services.dll	string get_Status() 0%	
698	-	15	workfluxor.dll	string get_Status() 0%	
288	-	52	workfluxor.services.dll	string get_Status() 0%	
316	-	49	workfluxor.services.dll	string get_Status() 0%	
342	-	50	workfluxor.services.dll	string get_Status() 0%	
343	-	48	workfluxor.services.dll	string get_Status() 0%	
157	-	45	workfluxor.services.dll	string get_Status() 0%	
261	-	38	workfluxor.services.dll	string get_StatusCode() 0%	
527	-	33	workfluxor.services.dll	string get_StatusCode() 0%	
559	-	34	workfluxor.services.dll	string get_StatusCode() 0%	
403	-	22	workfluxor.services.dll	string get_StatusCode() 0%	
434	-	30	workfluxor.services.dll	string get_StatusCode() 0%	
561	-	37	workfluxor.services.dll	string get_StatusCode() 0%	

Flaw Id	Module #	Class #	Module	Location	Fix By
480	-	23	workfluxor.services.dll	string get_StatusCode() 0%	
462	-	32	workfluxor.services.dll	string get_StatusCode() 0%	
38	-	36	workfluxor.services.dll	string get_StatusCode() 0%	
492	-	56	workfluxor.services.dll	string get_StatusCode() 0%	
184	-	58	workfluxor.services.dll	string get_StatusCode() 0%	
269	-	63	workfluxor.services.dll	string get_StatusCode() 0%	
438	-	31	workfluxor.services.dll	string get_StatusCode() 0%	
704	-	54	workfluxor.services.dll	string get_StatusCode() 0%	
295	-	55	workfluxor.services.dll	string get_StatusCode() 0%	
214	-	59	workfluxor.services.dll	string get_StatusCode() 0%	
441	-	57	workfluxor.services.dll	string get_StatusCode() 0%	
664	-	60	workfluxor.services.dll	string get_StatusDescription() 0%	
514	-	28	workfluxor.services.dll	string get_StatusDescription() 0%	
575	-	2	workfluxor.dll	string get_StatusID() 0%	
182	-	45	workfluxor.services.dll	string get_SubProductName() 0%	
674	-	46	workfluxor.services.dll	string get_SubProductName() 0%	
96	-	58	workfluxor.services.dll	string get_SubProductName() 0%	
486	-	53	workfluxor.services.dll	string get_SubProductName() 0%	
352	-	62	workfluxor.services.dll	string get_SubProfile() 0%	
669	-	62	workfluxor.services.dll	string get_SubSector() 0%	
517	-	53	workfluxor.services.dll	string get_SubSector() 0%	
107	-	59	workfluxor.services.dll	string get_SubSector() 0%	
235	-	15	workfluxor.dll	string get_Tags() 0%	
77	-	21	workfluxor.services.dll	string get_Tags() 0%	
69	-	21	workfluxor.services.dll	string get_Team() 0%	
407	-	62	workfluxor.services.dll	string get_TimeZone() 0%	

Flaw Id	Module #	Class #	Module	Location	Fix By
672	-	19	workfluxor.dll	string get_TrendChart() 0%	
759	-	3	workfluxor.dll	string get_Type() 0%	
263	-	45	workfluxor.services.dll	string get_UploadedBy() 0%	
264	-	45	workfluxor.services.dll	string get_UploadedDate() 0%	
67	-	21	workfluxor.services.dll	string get_UserEmail() 0%	
81	-	62	workfluxor.services.dll	string get_UserName() 0%	
670	-	14	workfluxor.dll	string get_Value() 0%	
237	-	45	workfluxor.services.dll	string get_VersionV() 0%	

Appendix B: Referenced Source Files

Id	Filename	Path
1	accountcontroller.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/controllers/
2	capacitycontroller.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/controllers/
3	commonvaluemodel.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/models/
4	docsearch.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor.docsearchsearvices/
5	encryption.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/helpers/
6	errorcontroller.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/controllers/
7	exportexcel.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/helpers/
8	feedbackcontroller.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/controllers/
9	forwardlookingutilizationentity.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor.services/entity/
10	homecontroller.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/controllers/
11	miscellaneouscontroller.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/controllers/
12	reccuranceapicontroller.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/controllers/
13	reportdetails.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor.services/entity/
14	reportscontroller.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/controllers/
15	repositorycontroller.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/controllers/
16	requestcontroller.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/controllers/
17	updaterequest.cshtml	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/obj/debug/aspnetcompilemerge/source/views/request/
18	usercontroller.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/controllers/
19	utility.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/helpers/
20	utilitycontroller.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/controllers/
21	utilizationforecastcontroller.cs	projects/workfluxorre/development/workfluxor-veracode/web project/workfluxor/controllers/

Appendix C: Referenced Classpaths

Id	Path
1	workfluxor_dll.Workfluxor.Controllers.RequestController.AddRequestEntity
2	workfluxor_dll.Workfluxor.Controllers.RequestController.UpdateRequestEntity
3	workfluxor_dll.Workfluxor.Controllers.UtilizationTrendChartData
4	workfluxor_dll.Workfluxor.Models.AcceptRejectRequestDeadlineModel
5	workfluxor_dll.Workfluxor.Models.CancelRequestModel
6	workfluxor_dll.Workfluxor.Models.ChangePasswordModel
7	workfluxor_dll.Workfluxor.Models.CheckNoOfDeliverablesModel
8	workfluxor_dll.Workfluxor.Models.ClientSideErrorModel
9	workfluxor_dll.Workfluxor.Models.CropImageModel
10	workfluxor_dll.Workfluxor.Models.FBTemplateDetailsFilterModel
11	workfluxor_dll.Workfluxor.Models.ForwardLookingUtilizationReportModel
12	workfluxor_dll.Workfluxor.Models.ReOpenRequestModel
13	workfluxor_dll.Workfluxor.Models.RequestDynamicsFilterModel
14	workfluxor_dll.Workfluxor.Models.StringValueModel
15	workfluxor_dll.Workfluxor.Models.UpdateAttachmentDetailModel
16	workfluxor_dll.Workfluxor.Models.UpdateRequestDeadlineModel
17	workfluxor_dll.Workfluxor.Models.UploadAttachmentModel
18	workfluxor_dll.Workfluxor.Models.UploadImageModel
19	workfluxor_dll.Workfluxor.Models.UtilizationDynamicsFilterModel
20	workfluxor_dll.Workfluxor.Models.UtilizationTrackerReportModel
21	workfluxor_services_dll.Workfluxor.Services.Entity.Attachments_Get_Data
22	workfluxor_services_dll.Workfluxor.Services.Entity.CountryDetails
23	workfluxor_services_dll.Workfluxor.Services.Entity.DesignationDetails
24	workfluxor_services_dll.Workfluxor.Services.Entity.DownlineUserData
25	workfluxor_services_dll.Workfluxor.Services.Entity.FbQuestionAndChoiceBySurveyIDEntity
26	workfluxor_services_dll.Workfluxor.Services.Entity.FbSurveyWithQuestionAndChoiceEntity
27	workfluxor_services_dll.Workfluxor.Services.Entity.FbUserResponseDetail
28	workfluxor_services_dll.Workfluxor.Services.Entity.ForwardLookingUtilizationEntity
29	workfluxor_services_dll.Workfluxor.Services.Entity.LanguageDetails
30	workfluxor_services_dll.Workfluxor.Services.Entity.MethodologyDetails
31	workfluxor_services_dll.Workfluxor.Services.Entity.PriorityDetails
32	workfluxor_services_dll.Workfluxor.Services.Entity.ProductTypeDetails
33	workfluxor_services_dll.Workfluxor.Services.Entity.ProjectTypeDetails
34	workfluxor_services_dll.Workfluxor.Services.Entity.RegionDetails
35	workfluxor_services_dll.Workfluxor.Services.Entity.RegionProjectTypeMappingEntity
36	workfluxor_services_dll.Workfluxor.Services.Entity.RegionSectorMappingDetailEntity
37	workfluxor_services_dll.Workfluxor.Services.Entity.RegionSectorMappingDetails
38	workfluxor_services_dll.Workfluxor.Services.Entity.ReportUtilizationDynamics

Id	Path
39	workfluxor_services_dll.Workfluxor.Services.Entity.ReportUtilizationDynamicsProduct
40	workfluxor_services_dll.Workfluxor.Services.Entity.ReportUtilizationDynamicsProject
41	workfluxor_services_dll.Workfluxor.Services.Entity.ReportUtilizationDynamicsRegion
42	workfluxor_services_dll.Workfluxor.Services.Entity.ReportUtilizationDynamicsRegionTrends
43	workfluxor_services_dll.Workfluxor.Services.Entity.ReportUtilizationDynamicsSector
44	workfluxor_services_dll.Workfluxor.Services.Entity.ReportUtilizationDynamicsSectorTrends
45	workfluxor_services_dll.Workfluxor.Services.Entity.RequestAttachmnetData
46	workfluxor_services_dll.Workfluxor.Services.Entity.RequestDetails
47	workfluxor_services_dll.Workfluxor.Services.Entity.RequestDetailsById_Get_Data
48	workfluxor_services_dll.Workfluxor.Services.Entity.RequestDynamicTrend
49	workfluxor_services_dll.Workfluxor.Services.Entity.RequestDynamicsByProductType
50	workfluxor_services_dll.Workfluxor.Services.Entity.RequestDynamicsByProjectType
51	workfluxor_services_dll.Workfluxor.Services.Entity.RequestDynamicsByRegion
52	workfluxor_services_dll.Workfluxor.Services.Entity.RequestDynamicsBySector
53	workfluxor_services_dll.Workfluxor.Services.Entity.RequestMISList
54	workfluxor_services_dll.Workfluxor.Services.Entity.SectorDetails
55	workfluxor_services_dll.Workfluxor.Services.Entity.SectorProductDetails
56	workfluxor_services_dll.Workfluxor.Services.Entity.ServiceTypeDetails
57	workfluxor_services_dll.Workfluxor.Services.Entity.StatusDetails
58	workfluxor_services_dll.Workfluxor.Services.Entity.SubProductTypeDetails
59	workfluxor_services_dll.Workfluxor.Services.Entity.SubSectorDetails
60	workfluxor_services_dll.Workfluxor.Services.Entity.UserDetails
61	workfluxor_services_dll.Workfluxor.Services.Entity.UserEntity
62	workfluxor_services_dll.Workfluxor.Services.Entity.UserList
63	workfluxor_services_dll.Workfluxor.Services.Entity.UserProjectMappingDetails

Appendix D: Dynamic Flaw Inventory

Rescan Status	Number of Flaws
All	0
New	0
Open and Reopened	0
Cannot Reproduce	0
Fixed	0