Zementis Enablement_ZENA Scan Report

Project Name	Zementis Enablement_ZENA
Scan Start	Tuesday, January 7, 2020 7:09:48 AM
Preset	Zementis_Enablement
Scan Time	22h:19m:28s
Lines Of Code Scanned	935719
Files Scanned	687
Report Creation Time	Sunday, January 12, 2020 6:56:58 AM
Online Results	http://DAESEC01/CxWebClient/ViewerMain.aspx?scanid=153289&projectid=523
Team	Zementis
Checkmarx Version	8.8.0.72 HF18
Scan Type	Subset
Source Origin	LocalPath
Density	1/100000 (Vulnerabilities/LOC)
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 v3.2	All
OWASP Top 10 2013	All
FISMA 2014	All
NIST SP 800-53	All
OWASP Top 10 2017	All
OWASP Mobile Top 10 2016	All

Excluded:

Uncategorized	None
Custom	None
PCI DSS v3.2	None
OWASP Top 10 2013	None
FISMA 2014	None

NIST SP 800-53	None
OWASP Top 10 2017	None
OWASP Mobile Top 10 2016	None

Results Limit

Results limit per query was set to 50

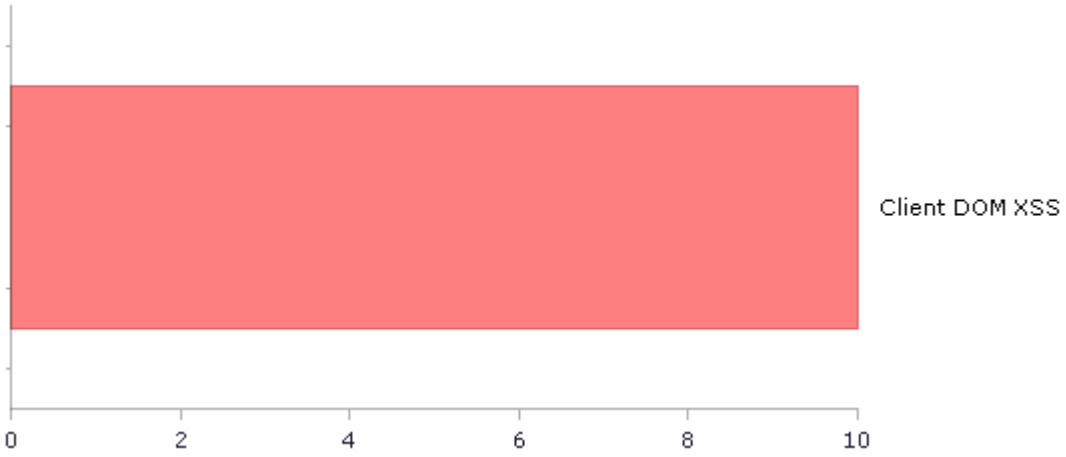
Selected Queries

Selected queries are listed in Result Summary

Result Summary

Most Vulnerable Files

Top 5 Vulnerabilities



Scan Summary - OWASP Top 10 2017

Further details and elaboration about vulnerabilities and risks can be found at: OWASP Top 10 2017

Category	Threat Agent	Exploitability	Weakness Prevalence	Weakness Detectability	Technical Impact	Business Impact	Issues Found	Best Fix Locations
A1-Injection*	App. Specific	EASY	COMMON	EASY	SEVERE	App. Specific	0	0
A2-Broken Authentication*	App. Specific	EASY	COMMON	AVERAGE	SEVERE	App. Specific	0	0
A3-Sensitive Data Exposure*	App. Specific	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	App. Specific	0	0
A4-XML External Entities (XXE)	App. Specific	AVERAGE	COMMON	EASY	SEVERE	App. Specific	0	0
A5-Broken Access Control	App. Specific	AVERAGE	COMMON	AVERAGE	SEVERE	App. Specific	0	0
A6-Security Misconfiguration	App. Specific	EASY	WIDESPREAD	EASY	MODERATE	App. Specific	0	0
A7-Cross-Site Scripting (XSS)*	App. Specific	EASY	WIDESPREAD	EASY	MODERATE	App. Specific	10	2
A8-Insecure Deserialization	App. Specific	DIFFICULT	COMMON	AVERAGE	SEVERE	App. Specific	0	0
A9-Using Components with Known Vulnerabilities*	App. Specific	AVERAGE	WIDESPREAD	AVERAGE	MODERATE	App. Specific	0	0
A10-Insufficient Logging & Monitoring	App. Specific	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	App. Specific	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 - 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	Business 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	0	0
A3-Cross-Site Scripting (XSS)*	EXTERNAL, INTERNAL, ADMIN USERS	AVERAGE	VERY WIDESPREAD	EASY	MODERATE	AFFECTED DATA AND SYSTEM	10	2
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	0	0
A6-Sensitive Data Exposure*	EXTERNAL, INTERNAL, ADMIN USERS, USERS BROWSERS	DIFFICULT	UNCOMMON	AVERAGE	SEVERE	EXPOSED DATA	0	0
A7-Missing Function Level Access Control	EXTERNAL, INTERNAL USERS	EASY	COMMON	AVERAGE	MODERATE	EXPOSED DATA AND FUNCTIONS	0	0
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 v3.2

Category	Issues Found	Best Fix Locations
PCI DSS (3.2) - 6.5.1 - Injection flaws - particularly SQL injection*	0	0
PCI DSS (3.2) - 6.5.2 - Buffer overflows	0	0
PCI DSS (3.2) - 6.5.3 - Insecure cryptographic storage*	0	0
PCI DSS (3.2) - 6.5.4 - Insecure communications	0	0
PCI DSS (3.2) - 6.5.5 - Improper error handling	0	0
PCI DSS (3.2) - 6.5.7 - Cross-site scripting (XSS)*	10	2
PCI DSS (3.2) - 6.5.8 - Improper access control*	0	0
PCI DSS (3.2) - 6.5.9 - Cross-site request forgery*	0	0
PCI DSS (3.2) - 6.5.10 - Broken authentication and session management	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 - FISMA 2014

Category	Description	Issues Found	Best Fix Locations
Access Control*	Organizations must limit information system access to authorized users, processes acting on behalf of authorized users, or devices (including other information systems) and to the types of transactions and functions that authorized users are permitted to exercise.	10	2
Audit And Accountability	Organizations must: (i) create, protect, and retain information system audit records to the extent needed to enable the monitoring, analysis, investigation, and reporting of unlawful, unauthorized, or inappropriate information system activity; and (ii) ensure that the actions of individual information system users can be uniquely traced to those users so they can be held accountable for their actions.	0	0
Configuration Management*	Organizations must: (i) establish and maintain baseline configurations and inventories of organizational information systems (including hardware, software, firmware, and documentation) throughout the respective system development life cycles; and (ii) establish and enforce security configuration settings for information technology products employed in organizational information systems.	0	0
Identification And Authentication*	Organizations must identify information system users, processes acting on behalf of users, or devices and authenticate (or verify) the identities of those users, processes, or devices, as a prerequisite to allowing access to organizational information systems.	0	0
Media Protection	Organizations must: (i) protect information system media, both paper and digital; (ii) limit access to information on information system media to authorized users; and (iii) sanitize or destroy information system media before disposal or release for reuse.	0	0
System And Communications Protection	Organizations must: (i) monitor, control, and protect organizational communications (i.e., information transmitted or received by organizational information systems) at the external boundaries and key internal boundaries of the information systems; and (ii) employ architectural designs, software development techniques, and systems engineering principles that promote effective information security within organizational information systems.	0	0
System And Information Integrity*	Organizations must: (i) identify, report, and correct information and information system flaws in a timely manner; (ii) provide protection from malicious code at appropriate locations within organizational information systems; and (iii) monitor information system security alerts and advisories and take appropriate actions in response.	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 - NIST SP 800-53

Category	Issues Found	Best Fix Locations
AC-12 Session Termination (P2)	0	0
AC-3 Access Enforcement (P1)	0	0
AC-4 Information Flow Enforcement (P1)	0	0
AC-6 Least Privilege (P1)	0	0
AU-9 Protection of Audit Information (P1)	0	0
CM-6 Configuration Settings (P2)	0	0
IA-5 Authenticator Management (P1)	0	0
IA-6 Authenticator Feedback (P2)	0	0
IA-8 Identification and Authentication (Non-Organizational Users) (P1)	0	0
SC-12 Cryptographic Key Establishment and Management (P1)	0	0
SC-13 Cryptographic Protection (P1)	0	0
SC-17 Public Key Infrastructure Certificates (P1)	0	0
SC-18 Mobile Code (P2)	0	0
SC-23 Session Authenticity (P1)*	0	0
SC-28 Protection of Information at Rest (P1)	0	0
SC-4 Information in Shared Resources (P1)	0	0
SC-5 Denial of Service Protection (P1)*	0	0
SC-8 Transmission Confidentiality and Integrity (P1)	0	0
SI-10 Information Input Validation (P1)*	0	0
SI-11 Error Handling (P2)	0	0
SI-15 Information Output Filtering (P0)*	10	2
SI-16 Memory Protection (P1)	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 - OWASP Mobile Top 10 2016

Category	Description	Issues Found	Best Fix Locations
M1-Improper Platform Usage	This category covers misuse of a platform feature or failure to use platform security controls. It might include Android intents, platform permissions, misuse of TouchID, the Keychain, or some other security control that is part of the mobile operating system. There are several ways that mobile apps can experience this risk.	0	0
M2-Insecure Data Storage	This category covers insecure data storage and unintended data leakage.	0	0
M3-Insecure Communication	This category covers poor handshaking, incorrect SSL versions, weak negotiation, cleartext communication of sensitive assets, etc.	0	0
M4-Insecure Authentication	This category captures notions of authenticating the end user or bad session management. This can include: -Failing to identify the user at all when that should be required -Failure to maintain the user's identity when it is required -Weaknesses in session management	0	0
M5-Insufficient Cryptography	The code applies cryptography to a sensitive information asset. However, the cryptography is insufficient in some way. Note that anything and everything related to TLS or SSL goes in M3. Also, if the app fails to use cryptography at all when it should, that probably belongs in M2. This category is for issues where cryptography was attempted, but it wasnt done correctly.	0	0
M6-Insecure Authorization	This is a category to capture any failures in authorization (e.g., authorization decisions in the client side, forced browsing, etc.). It is distinct from authentication issues (e.g., device enrolment, user identification, etc.). If the app does not authenticate users at all in a situation where it should (e.g., granting anonymous access to some resource or service when authenticated and authorized access is required), then that is an authentication failure not an authorization failure.	0	0
M7-Client Code Quality	This category is the catch-all for code-level implementation problems in the mobile client. That's distinct from server-side coding mistakes. This would capture things like buffer overflows, format string vulnerabilities, and various other code-level mistakes where the solution is to rewrite some code that's running on the mobile device.	0	0
M8-Code Tampering	This category covers binary patching, local resource modification, method hooking, method swizzling, and dynamic memory modification. Once the application is delivered to the mobile device, the code and data resources are resident there. An attacker can either directly modify the code, change the contents of memory dynamically, change or replace the system APIs that the application uses, or	0	0

	modify the application's data and resources. This can provide the attacker a direct method of subverting the intended use of the software for personal or monetary gain.		
M9-Reverse Engineering	This category includes analysis of the final core binary to determine its source code, libraries, algorithms, and other assets. Software such as IDA Pro, Hopper, otool, and other binary inspection tools give the attacker insight into the inner workings of the application. This may be used to exploit other nascent vulnerabilities in the application, as well as revealing information about back end servers, cryptographic constants and ciphers, and intellectual property.	0	0
M10-Extraneous Functionality	Often, developers include hidden backdoor functionality or other internal development security controls that are not intended to be released into a production environment. For example, a developer may accidentally include a password as a comment in a hybrid app. Another example includes disabling of 2-factor authentication during testing.	0	0

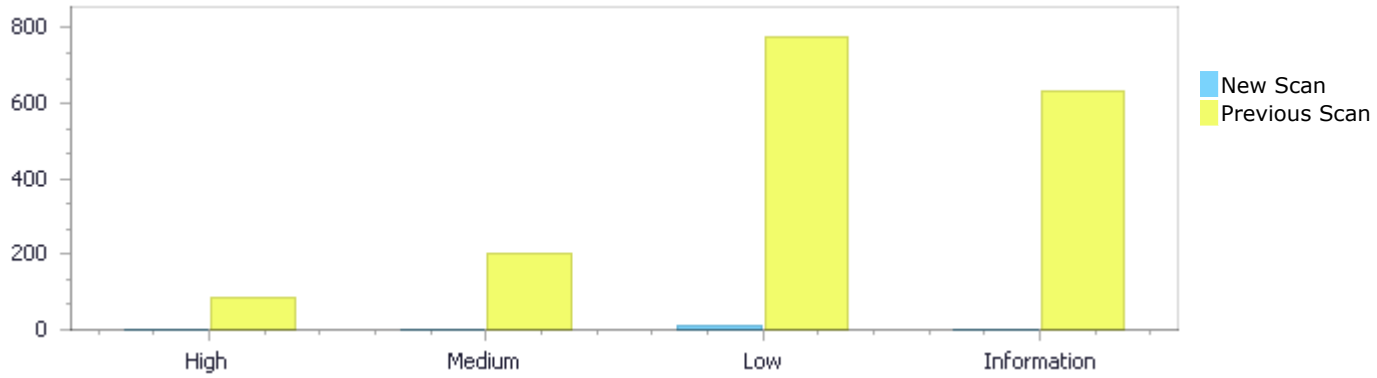
Scan Summary - Custom

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

Results Distribution By Status

Compared to project scan from 1/7/2020 7:35 AM

	High	Medium	Low	Information	Total
New Issues	0	0	0	0	0
Recurrent Issues	0	0	10	0	10
Total	0	0	10	0	10
Fixed Issues	86	204	768	631	1,689



Results Distribution By State

	High	Medium	Low	Information	Total
Confirmed	0	0	10	0	10
Not Exploitable	0	0	0	0	0
To Verify	0	0	0	0	0
Urgent	0	0	0	0	0
Proposed Not Exploitable	0	0	0	0	0
Total	0	0	10	0	10

Result Summary

Vulnerability Type	Occurrences	Severity
Client DOM XSS	10	High

Scan Results Details

Client DOM XSS

Query Path:
JavaScript\Cx\JavaScript High Risk\Client DOM XSS Version:1

Categories

PCI DSS v3.2: PCI DSS (3.2) - 6.5.7 - Cross-site scripting (XSS)
OWASP Top 10 2013: A3-Cross-Site Scripting (XSS)
FISMA 2014: Access Control
NIST SP 800-53: SI-15 Information Output Filtering (P0)
OWASP Top 10 2017: A7-Cross-Site Scripting (XSS)

Description

Client DOM XSS\Path 1:

Severity	Low
Result State	Confirmed
Online Results	http://DAESEC01/CxWebClient/ViewerMain.aspx?scanid=153289&projectid=523&pathid=404
Status	Recurrent

Method function at line 11892 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js gets user input for the data element. This element's value then flows through client-side code without being properly sanitized or validated and is eventually displayed to the user in e at line 43344 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js.This may enable a DOM XSS attack.

	Source	Destination
File	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js
Line	11893	43345
Object	data	create

Code Snippet

File Name zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js

Method }, e.prototype.disposeTemplate = function(e) {

```
....
11893.                  e.disposable.dispose(), e.disposable = null,
this.renderer.disposeTemplate(e.data), e.data = null
```

File Name zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js

Method function e(e, t, n, i, o, r) {

```
....
43345.                  this.codeEditor = e, this.themeService = r,
this.create(t, n, i, o)
```

Client DOM XSS\Path 2:

Severity	Low

Result State	Confirmed
Online Results	http://DAESEC01/CxWebClient/ViewerMain.aspx?scanid=153289&projectid=523&pathid=405
Status	Recurrent

Method function at line 11892 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js gets user input for the data element. This element's value then flows through client-side code without being properly sanitized or validated and is eventually displayed to the user in e at line 14713 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js.This may enable a DOM XSS attack.

	Source	Destination
File	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js
Line	11893	14714
Object	data	create

Code Snippet

File Name zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js

Method }, e.prototype.disposeTemplate = function(e) {

```
....
11893.                  e.disposable.dispose(), e.disposable = null,
this.renderer.disposeTemplate(e.data), e.data = null
```

File Name zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js

Method function e(e, t, n) {

```
....
14714.                      this.create(e, t, n)
```

Client DOM XSS\Path 3:

Severity	Low
Result State	Confirmed
Online Results	http://DAESEC01/CxWebClient/ViewerMain.aspx?scanid=153289&projectid=523&pathid=406
Status	Recurrent

Method function at line 11892 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js gets user input for the data element. This element's value then flows through client-side code without being properly sanitized or validated and is eventually displayed to the user in function at line 32295 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js.This may enable a DOM XSS attack.

	Source	Destination
File	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js
Line	11893	32296
Object	data	create

Code Snippet

File Name zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js

Method	}, e.prototype.disposeTemplate = function(e) {

```
....
11893.                  e.disposable.dispose(), e.disposable = null,
this.renderer.disposeTemplate(e.data), e.data = null
```

File Name	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js
Method	}, e.prototype.push = function(e) {

```
....
32296.                      return s.create(this, e)
```

Client DOM XSS\Path 4:

Severity	Low
Result State	Confirmed
Online Results	http://DAESEC01/CxWebClient/ViewerMain.aspx?scanid=153289&projectid=523&pathid=407
Status	Recurrent

Method function at line 2592 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js gets user input for the data element. This element's value then flows through client-side code without being properly sanitized or validated and is eventually displayed to the user in e at line 43344 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js.This may enable a DOM XSS attack.

	Source	Destination
File	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js
Line	2594	43345
Object	data	create

Code Snippet

File Name	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js
Method	}, e.prototype._traverse = function(e, t, n, o) {

```
....
2594.                      s = this._hashFn(e.data);
```

File Name	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js
Method	function e(e, t, n, i, o, r) {

```
....
43345.                  this.codeEditor = e, this.themeService = r,
this.create(t, n, i, o)
```

Client DOM XSS\Path 5:

Severity	Low
Result State	Confirmed
Online Results	http://DAESEC01/CxWebClient/ViewerMain.aspx?scanid=153289&projectid=5

	23&pathid=408
Status	Recurrent

Method function at line 2592 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js gets user input for the data element. This element's value then flows through client-side code without being properly sanitized or validated and is eventually displayed to the user in e at line 14713 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js.This may enable a DOM XSS attack.

	Source	Destination
File	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js
Line	2594	14714
Object	data	create

Code Snippet

File Name zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js

Method }, e.prototype._traverse = function(e, t, n, o) {

```
....
2594.                    s = this._hashFn(e.data);
```

File Name zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js

Method function e(e, t, n) {

```
....
14714.                    this.create(e, t, n)
```

Client DOM XSS\Path 6:

Severity	Low
Result State	Confirmed
Online Results	http://DAESEC01/CxWebClient/ViewerMain.aspx?scanid=153289&projectid=523&pathid=409
Status	Recurrent

Method function at line 2592 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js gets user input for the data element. This element's value then flows through client-side code without being properly sanitized or validated and is eventually displayed to the user in function at line 32295 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js.This may enable a DOM XSS attack.

	Source	Destination
File	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js
Line	2594	32296
Object	data	create

Code Snippet

File Name zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js

Method }, e.prototype._traverse = function(e, t, n, o) {

```
....
2594.                       s = this._hashFn(e.data);
```

File Name zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js

Method }, e.prototype.push = function(e) {

```
....
32296.                       return s.create(this, e)
```

Client DOM XSS\Path 7:

Severity	Low
Result State	Confirmed
Online Results	http://DAESEC01/CxWebClient/ViewerMain.aspx?scanid=153289&projectid=523&pathid=410
Status	Recurrent

Method function at line 2592 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js gets user input for the data element. This element's value then flows through client-side code without being properly sanitized or validated and is eventually displayed to the user in e at line 10399 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js.This may enable a DOM XSS attack.

	Source	Destination
File	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js
Line	2594	10402
Object	data	appendTo

Code Snippet

File Name zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js

Method }, e.prototype._traverse = function(e, t, n, o) {

```
....
2594.                       s = this._hashFn(e.data);
```

File Name zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js

Method function e(e, t, n) {

```
....
10402.                   this._onDidStart = new c.Emitter,
this._onDidChange = new c.Emitter, this._onDidReset = new c.Emitter,
this._onDidEnd = new c.Emitter, this.$e = i.$(".monaco-
sash").appendTo(e), s.isMacintosh && this.$e.addClass("mac"),
this.$e.on(d.EventType.MOUSE_DOWN, function(e) {
```

Client DOM XSS\Path 8:

Severity	Low
Result State	Confirmed

Online Results	http://DAESEC01/CxWebClient/ViewerMain.aspx?scanid=153289&projectid=523&pathid=411
Status	Recurrent

Method function at line 2592 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js gets user input for the data element. This element's value then flows through client-side code without being properly sanitized or validated and is eventually displayed to the user in e at line 10331 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js.This may enable a DOM XSS attack.

	Source	Destination
File	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js
Line	2594	10332
Object	data	create

Code Snippet

File Name zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js

Method }, e.prototype._traverse = function(e, t, n, o) {

```
....
2594.                        s = this._hashFn(e.data);
```

File Name zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js

Method function e(e, t) {

```
....
10332.                   this.options = t || Object.create(null),
l.mixin(this.options, u, !1), this.toUnbind = [], this.workedVal = 0,
this.progressBarBackground = this.options.progressBarBackground,
this.create(e)
```

Client DOM XSS\Path 9:

Severity	Low
Result State	Confirmed
Online Results	http://DAESEC01/CxWebClient/ViewerMain.aspx?scanid=153289&projectid=523&pathid=412
Status	Recurrent

Method function at line 2592 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js gets user input for the data element. This element's value then flows through client-side code without being properly sanitized or validated and is eventually displayed to the user in function at line 50244 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js.This may enable a DOM XSS attack.

	Source	Destination
File	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js
Line	2594	50262
Object	data	innerHTML

Code Snippet

File Name zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js
Method }, e.prototype._traverse = function(e, t, n, o) {

```
....
2594.                        s = this._hashFn(e.data);
```

File Name zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js
Method }, e.prototype.update = function(e) {

```
....
50262.                           p.innerHTML = f.getBaseLabel(d.resource)
+ "(" + d.startLineNumber + ", " + d.startColumn + "): ", p.title =
f.getPathLabel(d.resource), this._relatedDiagnostics.set(p, d);
```

Client DOM XSS\Path 10:

Severity	Low
Result State	Confirmed
Online Results	http://DAESEC01/CxWebClient/ViewerMain.aspx?scanid=153289&projectid=523&pathid=413
Status	Recurrent

Method function at line 2592 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js gets user input for the data element. This element's value then flows through client-side code without being properly sanitized or validated and is eventually displayed to the user in function at line 34347 of zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js.This may enable a DOM XSS attack.

	Source	Destination
File	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js	zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js
Line	2594	34363
Object	data	create

Code Snippet
File Name zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js
Method }, e.prototype._traverse = function(e, t, n, o) {

```
....
2594.                        s = this._hashFn(e.data);
```

File Name zmm/src/App/wwwroot/assets/monaco/vs/editor/editor.main.js
Method }, e.prototype._createInstance = function(e, t) {

```
....
34363.                   return p.push.apply(p, n), p.push.apply(p, r),
i.create.apply(null, p)
```

Client DOM XSS

Risk

What might happen

An attacker could use social engineering to cause a user to send the website engineered input, such as a URL with an engineered anchor, causing the browser to rewrite web pages. 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 web page includes data from user input (including the page URL). The user input is embedded in the page, 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 the webpage. Encoding should be context-sensitive. For example:
    - HTML encoding for HTML content.
    - HTML Attribute encoding for data output to attribute values.
    - JavaScript encoding for JavaScript.
3. Consider using the ESAPI4JS encoding library.

Source Code Examples

CSharp

For dynamically creating URLs in JavaScript, use the OWASP ESAPI4JS library:

```
window.location = ESAPI4JS.encodeForURL(input);
```

For creating dynamic HTML in JavaScript, use the OWASP ESAPI4JS library:

```
window.location = ESAPI4JS.encodeForURL(input);
```

Java

For dynamically creating URLs in JavaScript, use the OWASP ESAPI4JS library:

```
window.location = ESAPI4JS.encodeForURL(input);
```

For creating dynamic HTML in JavaScript, use the OWASP ESAPI4JS library:

```
window.location = ESAPI4JS.encodeForURL(input);
```

Scanned Languages

Language	Hash Number	Change Date
CSharp	0978409962023014	10/26/2018
JavaScript	3602822811217894	10/26/2018
VbScript	1349101913133594	10/26/2018
Python	0153239019201885	10/26/2018
Typescript	1939975091058023	10/26/2018
Common	0206692917308612	10/26/2018