Micro Focus WebInspect

Vulnerability

Web Application Assessment Report

Scan Name:	Site retest - Site: https://csvlhd21.esadev.dol.gov/olpdr/ - 1		
Policy:		Crawl Sessions:	4
Scan Date:	1/23/2020 10:22:33 PM	Vulnerabilities:	3
Scan Version:	18.20.178.0	Scan Duration:	2 minutes : 25 seconds
Scan Type:	Site Retest	Client:	IE

Server: https://csvlhd21.esadev.dol.gov:443

Medium Issues

Web Server Misconfiguration: Unprotected Directory (4731) View Description

CWE: 284,200

Kingdom: Environment

Page: https://csvlhd21.esadev.dol.gov:443/olpdr/scripts/unionSearchCtrl.js?%5c%22x%5c%22=12345

Request:

```
GET /olpdr/scripts/unionSearchCtrl.js?%5c%22x%5c%22=12345 HTTP/1.1
...TRUNCATED...
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 23 Jan 2020 22:24:47 GMT
X-Frame-Option...TRUNCATED...
```

Cross-Frame Scripting (11294) View Description

CWE: 352

Kingdom: Security Features

Page: https://csvlhd21.esadev.dol.gov:443/olpdr/scripts/unionSearchCtrl.js

Request:

```
GET /olpdr/scripts/unionSearchCtrl.js HTTP/1.1
Referer: https://csvlhd21.esadev.dol.gov/olpdr/
Host: csvlhd21.esadev.dol.gov
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-AscRawUrl: /olpdr/scripts/unionSearchCtrl.js
Pragma: no-cache
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:30.0) Gecko/20100101 Firefox/30.0
Connection: Keep-Alive
X-WIPP: AscVersion=18.20.178.0
X-Scan-Memo: Category="Audit.RetracePathFromRoot";
SID="1F5103F3DCD3134845B8113888851D26"; PSID="3C76525D3834840E163E598FA3F8F7DD";
SessionType="Crawl"; CrawlType="ScriptInclude"; AttackType="None";
OriginatingEngineID="00000000-0000-0000-0000-000000000000"; ThreadId="341";
ThreadType="AuditRetestTrace";
X-RequestManager-Memo: RequestorThreadIndex="0"; sid="817"; smi="0"; sc="1"; ID="004012d1-
1583-497e-be23-96195b9a224e";
X-Request-Memo: ID="2e11f3a2-1336-4314-81b2-7f4f29c40979"; sc="1"; ThreadId="341";
Cookie: CustomCookie=WebInspect0;JSESSIONID=000085EHiO31AhRVUbZlWPx91kQ:-1
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 23 Jan 2020 22:24:46 GMT
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=31536000; includeSubDomains
```

```
Cache-Control: private
Pragma: no-cache
Expires: -1
Last-Modified: Thu, 23 Jan 2020 14:51:26 GMT
Content-Length: 128379
X-XSS-Protection: 1; mode=block
Keep-Alive: timeout=10, max=100
Connection: Keep-Alive
Content-Type: application/javascript
Content-Language: en-US

app
                .controller(
                                'FileController',
                                function($scope, $location, $window, $anchorScroll, $timeout,$log) {

                                                $window.scrollTo(0, 0);
                                                $anchorScroll('custom-doc');

                                                $scope.printDiv = function() {
                                                                var printContents = document.getElementById
("custom-printpage").innerHTML;
                                                                var popupWin = $window.open('', '_blank',
'width=300,height=300');
                                                                popupWin.document.open();
                                                                popupWin.document
                                                                                .write('<html><head><link
rel="stylesheet" type="text/css" href="css/cssreset-min.css">      <link rel="stylesheet" type="text/css"
href="css/cssfonts-min.css">      <link rel="stylesheet" type="text/css" href="css/cssgrids-min.css"><link
rel="stylesheet" type="text/css" href="css/print.css"></head><body onload="window.print()">'
                                                                                                + printContents +
'</body></html>');
                                                                popupWin.document.close();
                                                };

                                });


var main = app
                .controller(
                                'unionSearchCtrl',
                                [
                                                '$scope',
                                                '$rootScope',
                                                '$http',par
        '$window',
                                                '$timeout',
                                                '$log',
                                                '$templateCache',
                                                '$filter',
                                                'uiGridConstants',
                                                'usSpinnerService',
                                                'ngDialog',
                                                'uiGridExporterConstants',
                                                'uiGridExporterService',
                                                function($scope, $rootScope, $http, $window,
$timeout, $log, $templateCache, $filter, uiGridConstants, usSpinnerService, ngDialog,
                                                                                uiGridExporterConstants,
uiGridExporterService) {

                                                                $rootScope.$watch('stateList', function
(newValue) {
```

```
        $scope.filerGrid.columnDefs[9].filter.selectOptions = newValue;

                                                        });
                                                        $rootScope.$watch('affAbbrList', function
(newValue) {

        $scope.filerGrid.columnDefs[3].filter.selectOptions = newValue;
                                                        });
                                                        $rootScope.$watch('desigNameList', function
(newValue) {

        $scope.filerGrid.columnDefs[5].filter.selectOptions = newValue;
                                                        });

                                                        $scope.closeTab = function(lmNo, event) {
                                                                var before =
$scope.filerGrid.gridApi.selection.getSelectedRows();

                                                                for (var i = 0; i < before.length; i++) {
                                                                        if (lmNo == before[i].lmNo) {

                                                                                $log.log("delete: " +
lmNo + " : " + before[i].lmNo);

                                        $scope.filerGrid.gridApi.selection.unSelectRow(before[i]);
                                                                        }
                                                                }
                                                                event.preventDefault();
                                                        };

                                                        $scope.openAttachmentLink = function(id) {


$window.open("/query/orgReport.do?rptId="+id+"&rptForm=attachment");
                                                                event.preventDefault();
                                                        }

                                                        $scope.loading = true;

                                                        $scope.removeAllFromCart = function() {
                                                                $scope.rptCart = [];
                                                        };

                                                         $scope.datePicker = {

                                                                options: {
                                                                   formatMonth: 'MM',
                                                                     startingDay: 1
                                                                },
                                                                format: "yyyy-MM-dd"

                                                             };

                                                          $scope.showDatePopup = [];

                                                                $templateCache.put('ui-grid/ui-grid-
date-filter',

                                                                   "<div class=\"ui-grid-filter-
container\" ng-repeat=\"colFilter in col.filters\" >" +


                                                                      "<input check-empty

type=\"text\" " +

                                                                     "uib-datepicker-popup=\"MMM
```

dd, yyyy\" " +
"datepicker
-options=\"datePicker.options\" " +
"datepicker-append-to-
body=\"true\" " +
"show-button-bar=\"false\""
+
"is-
open=\"showDatePopup[$index].opened\" class=\"input-sm form-control\"" +
"ng-
model=\"colFilter.term\" ng-attr-placeholder=\"{{colFilter.placeholder || aria.defaultFilterLabel}}\" " +
" aria-
label=\"{{colFilter.aria

...TRUNCATED...

Low Issues

HTML5: Form Validation Turned Off (11296) View Description

CWE: 20

Kingdom: Input Validation and Representation

Page: https://csvlhd21.esadev.dol.gov:443/olpdr/

Request:

```
GET /olpdr/ HTTP/1.1
Accept: */*
Pragma: no-cache
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:30.0) Gecko/20100101 Firefox/30.0
Host: csvlhd21.esadev.dol.gov
Connection: Keep-Alive
X-WIPP: AscVersion=18.20.178.0
X-Scan-Memo: Category="Audit.RetracePathFromRoot";
SID="3C76525D3834840E163E598FA3F8F7DD"; SessionType="ExternalAddedToCrawl";
CrawlType="None"; AttackType="None"; OriginatingEngineID="00000000-0000-0000-0000-
000000000000"; ThreadId="341"; ThreadType="AuditRetestTrace";
X-RequestManager-Memo: RequestorThreadIndex="0"; sid="817"; smi="0"; sc="1"; ID="6075d8ff-af57-
4bb9-ad8e-346ad50c0a0c";
X-Request-Memo: ID="05c70883-a1b2-4ddd-8686-de426ab31e5c"; sc="1"; ThreadId="341";
Cookie: CustomCookie=WebInspect0
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 23 Jan 2020 22:22:40 GMT
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=31536000; includeSubDomains
Cache-Control: private
Pragma: no-cache
Expires: -1
X-XSS-Protection: 1; mode=block
Content-Length: 60158
Set-Cookie: JSESSIONID=000085EHiO31AhRVUbZlWPx91kQ:-1; Path=/; Secure;
HttpOnly;HttpOnly;Secure
Keep-Alive: timeout=10, max=100
Connection: Keep-Alive
```

```
Content-Type: text/html; charset=UTF-8
Content-Language: en-US

...TRUNCATED...orts are listed to the right.</div> -->
                        <form novalidate name="paymentQueryForm" id="paymentQueryForm">
        ...TRUNCATED...w">
                <div class="col-xs-2 well">

                        <form novalidate name="paymentQueryForm" id="paymentQueryForm">
...TRUNCATED...s="row">
                <div class="col-xs-2 well">
                        <form novalidate name="paymentQueryForm" id="paymentQueryForm">
        ...TRUNCATED...
```

Appendix (Check Descriptions)

Web Server Misconfiguration: Unprotected Directory (4731)

Summary

A directory was discovered that contains an object referenced in a post request or query string, and which has a name that could easily be guessed by an attacker. The primary danger from an attacker discovering this directory would arise from the information he could gather from its contents, such as what language was used to code the web application. Recommendations include restricting access to important directories or files by adopting a "need to know" requirement for both the document and server root, and turning off features such as Automatic Directory Listings that provide information that could be utilized by an attacker when formulating or conducting an attack.

Execution

This check determines the parent directory of an object referenced in either a post request or query string, and ascertains whether the name of the directory could lead to security issues.

Fix

For Security Operations:

One of the most important aspects of web application security is to restrict access to important files or directories only to those individuals who actually need to access them. The harder you make it for an attacker to access information about your web application, the more likely it is that he will simply find an easier target. Use the following recomendations to improve the security of your web application.

- Restrict access to important files or directories only to those who actually need it. More information about implementing secure authentication schemes is listed below.
- Enforce consistent authentication across your entire application. Ensure authentication is applied to the entire directory structure, including subdirectories.
- Limit server-side includes only to trusted sources with adequate permissions.
- Ensure that files containing sensitive information are not left publicly accessible.
- Do not follow standard naming procedures for hidden directories. For example, don't create a hidden directory called "cgi" that contains cgi scripts. Obvious directory names are just that...readily guessed by an attacker.
- Enforce a least privileges access policy.
- Do not use robots.txt files, as most search engines and crawling/spidering tools do not honor them.

For Development:

Unless you are actively involved with implementing the web application server, there is not a wide range of available solutions to prevent problems that can occur from an attacker discovering underlying architectural information about your application. The nature of a web site is to be publicly accessible, which means the directory structure will also be so. Primarily, this problem will be resolved by the web application server administrator. However, there are certain actions you can take that will

help to secure your web application and make it harder for an attacker to conduct a successful attack.

- Ensure that files containing sensitive information are not left publicly accessible, or that comments left inside files do not reveal the locations of directories best left confidential.
- Do not reveal information in pathnames that are publicly displayed. Do not include drive letters or directories outside of the web document root in the pathname when a file must call another file on the web server. Use pathnames that are relative to the current directory or the webroot.

For QA:

For reasons of security, it is important to test the web application not only from the perspective of a normal user, but also from that of a malicious one. Whenever possible, adopt the mindset of an attacker when testing your web application for security defects. Access your web application from outside your firewall or IDS. Utilize Google or another search engine to ensure that searches for vulnerable files or directories do not return information regarding your web application. For example, an attacker will utilize a search engine, and search for directory listings such as the following: 'index of / cgi-bin'. Make sure that your directory structure is not obvious, and that only files that are necessary are capable of being accessed.

Reference

Classifications

CWE-284: Access Control (Authorization) Issues
http://cwe.mitre.org/data/definitions/284.html

CWE-200: Information Exposure
http://cwe.mitre.org/data/definitions/200.html

Kingdom: Environment
https://vulncat.fortify.com/

Cross-Frame Scripting (11294)

Summary

A Cross-Frame Scripting (XFS) vulnerability can allow an attacker to load the vulnerable application inside an HTML iframe tag on a malicious page. The attacker could use this weakness to devise a Clickjacking attack to conduct phishing, frame sniffing, social engineering or Cross-Site Request Forgery attacks.

Clickjacking
The goal of a Clickjacking attack is to deceive the victim (user) into interacting with UI elements of the attacker's choice on the target web site without their knowledge and then executing privileged functionality on the victim's behalf. To achieve this goal, the attacker must exploit the XFS vulnerability to load the attack target inside an iframe tag, hide it using Cascading Style Sheets (CSS) and overlay the phishing content on the malicious page. By placing the UI elements on the phishing page so they overlap with those on the page targeted in the attack, the attacker can ensure that the victim must interact with the UI elements on the target page not visible to the victim.

WebInspect has detected a response containing one or more forms that accept user input but is missing XFS protection.

Execution

Create a test page containing an HTML iframe tag whose src attribute is set to https://csvlhd21.esadev.dol.gov:443/olpdr/scripts/unionSearchCtrl.js. Successful framing of the target page indicates that the application is susceptibile to XFS.

Note that WebInspect will report only one instance of this check across each host within the scope of the scan. The other visible pages on the site may, however, be vulnerable to XFS as well and therefore should be protected against it with an appropriate fix.

Implication

A Cross-Frame Scripting weakness could allow an attacker to embed the vulnerable application inside an iframe. Exploitation of this weakness could result in:

Hijacking of user events such as keystrokes
Theft of sensitive information
Execution of privileged functionality through combination with Cross-Site Request Forgery attacks

Fix

The Content Security Policy (CSP) frame-ancestors directive obsoletes the X-Frame-Options header. Both provide for a policy-based mitigation technique against cross-frame scripting vulnerabilities. The difference is that while the X-Frame-Options technique only checks against the top-level document's location, the CSP frame-ancestors header checks for conformity from all ancestors.

If both CSP frame-ancestors and X-Frame-Options headers are present and supported, the CSP directive will prevail. WebInspect recommends using both CSP frame-ancestors and X-Frame-Options headers as CSP is not supported by Internet Explorer and many older versions of other browsers.

In addition, developers must also use client-side frame busting JavaScript as a protection against XFS. This will enable users of older browsers that do not support the X-Frame-Options header to also be protected from Clickjacking attacks.

X-Frame-Options
Developers can use this header to instruct the browser about appropriate actions to perform if their site is included inside an iframe.Developers must set the X-Frame-Options header to one of the following permitted values:

- DENY

Deny all attempts to frame the page

- SAMEORIGIN

The page can be framed by another page only if it belongs to the same origin as the page being framed

- ALLOW-FROM origin

Developers can specify a list of trusted origins in the origin attribute. Only pages on origin are permitted to load this page inside an iframe

Content-Security-Policy: frame-ancestors

Developers can use the CSP header with the frame-ancestors directive, which replaces the X-Frame-Options header, to instruct the browser about appropriate actions to perform if their site is included inside an iframe. Developers can set the frame-ancestors attribute to one of the following permitted values:

- 'none'

Equivalent to "DENY" - deny all attempts to frame the page

- 'self'

Equivalent to "SAMEORIGIN" - the page can be framed by another page only if it belongs to the same origin as the page being framed

- <host-source>

Equivalent to "ALLOW-FROM" - developers can specify a list of trusted origins which maybe host name or IP address or URL scheme. Only pages on this list of trusted origin are permitted to load this page inside an iframe

- <scheme-source>

Developers can also specify a schema such as http: or https: that can frame the page.

Reference

Frame Busting:
Busting Frame Busting: A Study of Clickjacking Vulnerabilities on Popular Sites

OWASP: Busting Frame Busting

OWASP:
Clickjacking

Content-Security-Policy (CSP)
CSP: frame-ancestors

Specification:
Content Security Policy Level 2
X-Frame-Options IETF Draft

Server Configuration:
IIS
Apache, nginx

HP 2012 Cyber Security Report
The X-Frame-Options header - a failure to launch

Classifications

CWE-352: Cross-Site Request Forgery (CSRF)
http://cwe.mitre.org/data/definitions/352.html

Kingdom: Security Features
https://vulncat.fortify.com/

HTML5: Form Validation Turned Off (11296)

Summary

HTML5 provides a convenient way to add client-side input form field validation by simply including a "required" attribute for required fields or "fieldtype" attribute for common input types or, more granularly, by enabling users to specify a data type-sensitive context that allows pattern-based validation. Developers can supply a customizable "pattern" attribute that checks the input against a regular expression. However, this validation gets disabled when adding a "novalidate" attribute on a form field or a "formnovalidate" attribute on a submit input form field. This check detects whether the "novalidate" or "formnovalidate" attributes are used which may lead to a security vulnerability depending on how securely the application is developed.

Implication

HTML forms with disabled validation can potentially expose the server to numerous types of attacks. Unchecked input is the root cause of vulnerabilities like cross-site scripting and SQL injection.

Fix

While it is much more important to validate input on the server side, validation on the client side adds another layer of protection and makes the process of completing HTML forms more user-friendly and responsive. Avoid using the "novalidate" or "formnovalidate" attributes alone unless compensating methods of validation are being used in their place, and always implement complementary form field validation from the application server; never relying solely on client-side validation routines in the browser.

Reference

HTML5 <form> novalidate Attribute
http://www.w3schools.com/html5/att_form_novalidate.asp

HTML5 <input> formnovalidate Attribute
http://www.w3schools.com/html5/att_input_formnovalidate.asp

Classifications

CWE-20: Improper Input Validation Secondary
http://cwe.mitre.org/data/definitions/20.html

Kingdom: Input Validation and Representation
https://vulncat.fortify.com/