|  |  |  |
| --- | --- | --- |
|  | |  |
|  | | Requirements Document |
|  | |  |
|  | Cross-site scripting detector  **Version: 1.0**  **Last Revised:** September 29, 2010  **Author: Maksym Sukhovarov** | |

Table of Contents

[Feature Overview 6](#_Toc273952291)

[1 Terminology 6](#_Toc273952292)

[2 Product Requirements 6](#_Toc273952293)

[2.1 Java script code detection and global functions detection 6](#_Toc273952294)

[2.2 XML DOM Elements 7](#_Toc273952295)

[2.3 HTML-tags detection 8](#_Toc273952296)

[2.4 VB script code detection 10](#_Toc273952297)

[2.5 Signs detection 11](#_Toc273952298)

[2.6 Obfuscation code detection 12](#_Toc273952299)

[3 Functional Area 15](#_Toc273952300)

Document Version History

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Revision No.** | **Date** | **Description of Change** |
| Maksym Sukhovarov | 1.0 | 09/23/2010 | First revision |
| Maksym Sukhovarov | 1.0 | 09/29/2010 | Reordered requirements, removed events for video tag and sing detections was added |
| Anatoliy Lokshin | 1.1 | 10/04/2010 | Grammar correction. |
|  |  |  |  |

Approvers

|  |  |  |
| --- | --- | --- |
| **Name** | **Title** | **Date Approved** |
|  |  |  |
|  |  |  |
|  |  |  |

Reviewers

| **Name** | **Title** | **Date Reviewed** |
| --- | --- | --- |
| **Kyle Quest** |  |  |
|  |  |  |
|  |  |  |

References

|  |  |  |
| --- | --- | --- |
| **Document Name** | **Author** | **Location** |
| |  | | --- | | MetraTech Security Framework Specification | | Kyle Quest | <http://seceng.metratech.com/gf/download/docmanfileversion/20/78/MtSecurityFrameworkSpec.doc> |
| Cross Site Scripting Attacks | Kyle Quest | <http://seceng.metratech.com/gf/download/docmanfileversion/3/48/CrossSiteScriptingAttacks.pdf> |
| HTML Code Injection and Cross-site scripting Understanding the cause and effect of CSS (XSS) Vulnerabilities | <http://www.technicalinfo.net/papers/CSS.html> | <http://seceng.metratech.com/gf/download/docmanfileversion/40/107/HTML_Code_Injection_and_Cross_site_scripting.docx> |
| Cross Site Scripting Exploits And Defense | Jeremiah Grossman, Robert Hansen, Petko D. Petkov, Anton Rager, Seth Fogie | <http://seceng.metratech.com/gf/download/docmanfileversion/12/57/Cross_Site_Scripting_Attacks_and_Defense.pdf> |
| Client-Side Detection of XSS Worms by Monitoring Payload Propagation | Department of Computer Science University of California, Davis | <http://seceng.metratech.com/gf/download/docmanfileversion/38/105/Client-SideDetectionofXSSWorms.pdf> |
| XSSDS: Server-side Detection of Cross-site Scripting Attacks | Martin Johns, Bjorn Engelmann, and Joachim Posegga | <http://seceng.metratech.com/gf/download/docmanfileversion/39/106/XssDS--Server-SideDetectiobofCross-SiteScriptingAttacks.pdf> |

References to Encoders

|  |  |
| --- | --- |
| **Page Name** | **Location** |
| URL Decoder/Encoder | <http://meyerweb.com/eric/tools/dencoder/> |
| Base 64 Encoder | <http://www.opinionatedgeek.com/dotnet/tools/Base64Encode/Default.aspx> |
| Escape, encodeURI, Url Encode | <http://www.the-art-of-web.com/javascript/escape/> |
| Unicode Converter / Encoder | <http://www.pinnacledisplays.com/unicode-converter.htm> |
| Script Ebcoding | <http://scriptasylum.com/tutorials/encode-decode.html> |
| Java Script description global functions | <http://www.w3schools.com/jsref/jsref_obj_global.asp> |
| VbScript description global functions | <http://www.w3schools.com/vbscript/func_getobject.asp> |
| HTML 5 tags description | <http://www.w3schools.com/html5/html5_reference.asp> |
| HTML 5 event attributes | <http://www.w3schools.com/html5/html5_ref_eventattributes.asp> |

References to descriptions XSS attacks

|  |  |
| --- | --- |
| **Page Name** | **Location** |
| Online Script obfuscation | <http://www.crypo.com/eng_jse.php> |
| Advanced JavaScript obfuscation (or why signature scanning is a failure) | <http://isc.sans.edu/diary.html?storyid=6142> |
| Advanced obfuscated JavaScript analysis | <http://isc.sans.edu/diary.html?storyid=4246> |
| Using event for XSS attack | <http://wiki.auditory.ru/XSS-%D0%B0%D1%82%D0%B0%D0%BA%D0%B8> |
| Drawing similarities between email and web attacks | <http://community.websense.com/blogs/securitylabs/archive/2010/06/14/drawing-similarities-between-email-and-web-attacks.aspx> |
| Hiding scripts | <http://securitylabs.websense.com/content/Blogs/3408.aspx> |

# Feature Overview

The Cross-site Scripting (XSS) detector is responsible for providing the detecting of XSS attack.

# Terminology

Requirements that are labeled with an R (such as R-300) are Mandatory (or Required) for the targeted release.

Requirements that are labeled with an O (such as O-300) are Optional for the targeted release.

It is left to Engineering’s discretion on whether these can be met for the targeted release but it is imperative that Engineering take these into consideration when making any design or infrastructure decisions. This holds good for any requirement that might be listed under “Future Requirements” section as well.

**(R-1)** It is a Mandatory requirement that the infrastructure developed shall support the ‘feasibility’ of all feature requirements listed in the document – though the delivery of the features themselves could be scheduled for a later time.

# Product Requirements

If XSS detector found an attack then it should be throw an exception.

The detector should be detected following constructions:

* Java script code;
* XML DOM elements
* Using some dangerous html tags;
* VB script code;
* Obfuscation code;

## Java script code detection and global functions detection

Any dangerous java script code must be detected. For example:

* 1. [**decodeURI**(string)](http://www.w3schools.com/jsref/jsref_decodeuri.asp);
  2. [**decodeURIComponent**(string)](http://www.w3schools.com/jsref/jsref_decodeuricomponent.asp);
  3. [**encodeURI**(string)](http://www.w3schools.com/jsref/jsref_encodeuri.asp);
  4. [**encodeURIComponent**(string)](http://www.w3schools.com/jsref/jsref_encodeuricomponent.asp);
  5. [**escape**(string)](http://www.w3schools.com/jsref/jsref_escape.asp);
  6. [**eval**(string)](http://www.w3schools.com/jsref/jsref_eval.asp);
  7. [**isFinite**(value)](http://www.w3schools.com/jsref/jsref_isfinite.asp);
  8. [**isNaN**(value)](http://www.w3schools.com/jsref/jsref_isnan.asp);
  9. [**Number**(object)](http://www.w3schools.com/jsref/jsref_number.asp);
  10. [**parseFloat**(string)](http://www.w3schools.com/jsref/jsref_parsefloat.asp);
  11. [**parseInt**(string, radix)](http://www.w3schools.com/jsref/jsref_parseint.asp);
  12. [**String**(object)](http://www.w3schools.com/jsref/jsref_string.asp);
  13. [**unescape**(string)](http://www.w3schools.com/jsref/jsref_unescape.asp)

**List of these elements should be contained in a configuration file.**

## XML DOM Elements

Any XML DOM elements must be detected, for example:

1. *document.write(document.links[0].id);*
2. *document.write(document.domain);*
3. *var doc=document.open("text/html","replace");  var txt="<html><body>Learning about the HTML DOM is fun!</body></html>";  doc.write(txt);*
4. alert(x.innerHTML);
5. document.getElementById("htmldom").href;
6. var p=window.createPopup();var pbody=p.document.body;
7. window.scrollBy(100,100);
8. window.location.assign("http://www.w3schools.com")

**List of these elements should be contained in a configuration file.**

## HTML-tags detection

Some HTML-tags that can be used to XSS attack, so the following list of tags should be defined as an attack:

* + Comments (<!-- );
  + <a>;
  + <aplet>;
  + <area>;
  + <audio>;
  + <base>;
  + <blockquote>;
  + <body>; (contain a lot of event attributes)
  + <button>;
  + <command>;
  + <del>
  + <embed>;
  + <form>;
  + <html>;
  + <iframe>;
  + <img>;
  + <input>;
  + <ins>;
  + <keygen>;
  + <link>;
  + <meta>;
  + <object>;
  + <script>;
  + <source>;
  + <video>

**Black list of these elements should be contained in a configuration file.**

Other tags could be use, but if tags contain any events then this tags should be identifying.

Below is a list of events that can use in html tags.

1. Form events attributes:

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| onblur | *script* | Script to be run when an element loses focus |
| onchange | *script* | Script to be run when an element changes |
| oncontextmenu**New** | *script* | Script to be run when a context menu is triggered |
| onfocus | *script* | Script to be run when an element gets focus |
| onformchange**New** | *script* | Script to be run when a form changes |
| onforminput**New** | *script* | Script to be run when a form gets user input |
| oninput**New** | *script* | Script to be run when an element gets user input |
| oninvalid**New** | *script* | Script to be run when an element is invalid |
| onreset | *script* | Script to be run when a form is reset Not supported in HTML 5 |
| onselect | *script* | Script to be run when an element is selected |
| onsubmit | *script* | Script to be run when a form is submitted |

1. Keyboard events attributes:

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| onkeydown | *script* | Script to be run when a key is pressed |
| onkeypress | *script* | Script to be run when a key is pressed and released |
| onkeyup | *script* | Script to be run when a key is released |

1. Mouse events:

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| onclick | *script* | Script to be run on a mouse click |
| ondblclick | *script* | Script to be run on a mouse double-click |
| ondrag**New** | *script* | Script to be run when an element is dragged |
| ondragend**New** | *script* | Script to be run at the end of a drag operation |
| ondragenter**New** | *script* | Script to be run when an element has been dragged to a valid drop target |
| ondragleave**New** | *script* | Script to be run when an element leaves a valid drop target |
| ondragover**New** | *script* | Script to be run when an element is being dragged over a valid drop target |
| ondragstart**New** | *script* | Script to be run at the start of a drag operation |
| ondrop**New** | *script* | Script to be run when dragged element is being dropped |
| onmousedown | *script* | Script to be run when a mouse button is pressed |
| onmousemove | *script* | Script to be run when the mouse pointer moves |
| onmouseout | *script* | Script to be run when the mouse pointer moves out of an element |
| onmouseover | *script* | Script to be run when the mouse pointer moves over an element |
| onmouseup | *script* | Script to be run when a mouse button is released |
| onmousewheel**New** | *script* | Script to be run when the mouse wheel is being rotated |
| onscroll**New** | *script* | Script to be run when an element's scrollbar is being scrolled |

**List of these elements should be contained in a configuration file.**

## VB script code detection

Any dangerous VB Script code must be detected, for example:

1. Function **CanDeliver(Dt)** CanDeliver = (CDate(Dt) - Now()) > 2 End Function
2. [**CreateObject**](http://www.w3schools.com/vbscript/func_createobject.asp)(string);
3. [**Eval**](http://www.w3schools.com/vbscript/func_eval.asp)(*expression*);
4. [**GetLocale**](http://www.w3schools.com/vbscript/func_getlocale.asp)(pathname, class);
5. [**GetObject**](http://www.w3schools.com/vbscript/func_getobject.asp)();
6. [**GetRef**](http://www.w3schools.com/vbscript/func_getref.asp)(procname);
7. [**InputBox**](http://www.w3schools.com/vbscript/func_inputbox.asp)(prompt[,title][,default][,xpos][,ypos][,helpfile,context]);
8. [**IsEmpty**](http://www.w3schools.com/vbscript/func_isempty.asp)(expression);
9. [**IsNull**](http://www.w3schools.com/vbscript/func_isnull.asp)(expression);
10. [**IsNumeric**](http://www.w3schools.com/vbscript/func_isnumeric.asp)(expression);
11. [**IsObject**](http://www.w3schools.com/vbscript/func_isobject.asp)(expression);
12. [**LoadPicture**](http://www.w3schools.com/vbscript/func_loadpicture.asp)(picturename);
13. [**MsgBox**](http://www.w3schools.com/vbscript/func_msgbox.asp)(prompt[,buttons][,title][,helpfile,context]);
14. [**RGB**](http://www.w3schools.com/vbscript/func_rgb.asp)(red,green,blue);
15. [**Round**](http://www.w3schools.com/vbscript/func_round.asp)(expression[,numdecimalplaces]);
16. [**ScriptEngine**](http://www.w3schools.com/vbscript/func_scriptengine.asp);
17. [**ScriptEngineBuildVersion**](http://www.w3schools.com/vbscript/func_scriptengine.asp);
18. [**ScriptEngineMajorVersion**](http://www.w3schools.com/vbscript/func_scriptengine.asp);
19. [**ScriptEngineMinorVersion**](http://www.w3schools.com/vbscript/func_scriptengine.asp);
20. [**SetLocale**](http://www.w3schools.com/vbscript/func_setlocale.asp)(short string or hex value or decimal value);
21. [**TypeName**](http://www.w3schools.com/vbscript/func_typename.asp)(varname);
22. [**VarType**](http://www.w3schools.com/vbscript/func_vartype.asp)(varname);

**List of these elements should be contained in a configuration file.**

## Signs detection

Signs in conjunction with the escape characters can carry an attack, so they should be identified. For example:

1. ';';
2. ‘/’;
3. ‘?’;
4. ‘:’;
5. ‘@’;
6. ‘=’;
7. ‘&’;
8. ‘<’;
9. ‘>’;
10. ’ ” ’;
11. ‘#’;
12. ‘{’;
13. ‘}’;
14. ‘/’;
15. ‘\’;
16. ‘|’;
17. ‘^’;
18. ‘~’;
19. ‘[’;
20. ‘]’;
21. ‘ ’ ’;
22. ‘%’.

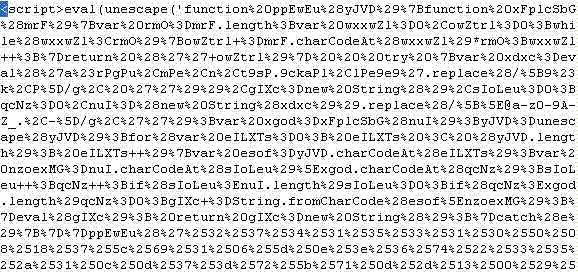
## Obfuscation code detection

The detector must identify such sequences, which can hide the javascript or vbscript code. It’s possible to formulate the following rules:

1. URL encoding (percent encoding), for example:

*“%3Csrcipt%3Ealert('xss')%3B%3C%2Fsrcipt%3E*”;

*“%3Cscript%3Ealert('%D0%9F%D1%80%D0%B8%D0%B2%D0%B5%D1%82')%3B%3C%2Fscript%3E*”;



1. Encoding binary data Base64 group
   1. Base64 – end of string is **any symbol** or ‘**=**’ or “**==**”;
   2. MIME Base64 – end of string is **any symbol** or ‘**+**’ or “**+/**”;
   3. Base64 for URL – end of string is **any symbol** or ‘**\***’ or “**\*-**“;
   4. Radix 64 – is identical to the "Base64" encoding described from MIME, with the addition of an optional 24-bit [CRC](http://en.wikipedia.org/wiki/Cyclic_redundancy_check) checksum, for example:

“*http://example.com/mail?redirect-after-login=data%3Atext/html%3Bbase64%2CPHNjcmlwdD4KYWxlcnQoJ1hTUycpOwo8L3NjcmlwdD4%3D*”.

1. HTML encoding, for example:

*“&lt;script&gt;alert(&#39;hi&#39;);&lt;/script&gt;*”;

1. Javascript escaping, for example:

*“%3Cscript%3Ealert%28%27%u041F%u0440%u0438%u0432%u0435%u0442%27%29%3B%3C/script%3E*”;

1. Unicode encoding, for example:

*“&#60;&#115;&#99;&#114;&#105;&#112;&#116;&#62;&#97;&#108;&#101;&#114;&#116;&#40;&#39;&#1055;&#1088;&#1080;&#1074;&#1077;&#1090;&#39;&#41;&#59;&#60;&#47;&#115;&#99;&#114;&#105;&#112;&#116;&#62;*” and was encoded:

*“<script>alert('Привет');</script>”*

1. Hiding code by logic expressions, for example:
2. rgvij=(0.2e1>=4e1?.9075:"i"+"f");
3. document[“write”](“text to print”);
4. aaa=(((0x4435,7.)>=(.61,9.12e2)?(1,4.033e3):(266,7.1e1)),((0x97<=.1?7.616e3:2.176e3),(.39<8e0?document:2032)));

And try to detect other any encoding code.

**If it possible define this rule definition in a configuration file.**

# Functional Area

| Requirement | Story | Constraints |
| --- | --- | --- |
| R-300 – Mandatory requirement | JavaScript code detection and global functions detection |  |
| R-301 – Mandatory requirement | XML DOM elements detection |  |
| R-302 – Mandatory requirement | HTML-tags detection (black list of tags) |  |
| R-303 – Mandatory requirement | HTML events attributes detection (part 1):   * Form events attributes; * Keyboard events attributes; * Mouse events. |  |
| R-304 – Mandatory requirement | VB script code detection |  |
| R-305 – Mandatory requirement | Obfuscation code detection (part 1):   * url encoding; * html encoding; * javascript escaping; * Unicode encoding. |  |
| R-306 – Mandatory requirement | Assignment of weights for the identification of a possible attack and to develop a criterion for determining the probability of XSS attack. Also use sings detection. |  |
| R-307 – Mandatory requirement | XSS detector supports a rule set to determine decoding operations required.  Such an engine is necessary to create a Processor subsystem.  The composite information processing engine must provide a possibility of defining rules for processing input stream.  For that purpose, sequence and branching rules for including appropriate information-processing engines can be used. |  |
| O-308 – Optional requirement | Obfuscation code detect (part 2) - Hiding code by logic expressions |  |
| O-309 – Mandatory requirement | Obfuscation code detect (part 3) - BASE64 encoding |  |