|  |  |  |
| --- | --- | --- |
|  | |  |
|  | | Engineering Design Document |
|  | |  |
|  | Decoder subsystem  **Version: 1.0**  **Last Revised:**  **Author: Anatoliy Lokshin** | |

Table of Contents

[1 System Overview 4](#_Toc278273342)

[2 Assumptions and Dependencies 4](#_Toc278273343)

[3 Technical Design 4](#_Toc278273344)

[3.1 Class diagram and description 4](#_Toc278273345)

[3.2 High-Level Design / Tasks 6](#_Toc278273346)

[4 Error list 6](#_Toc278273347)

[5 Outstanding Issues 7](#_Toc278273348)

Document Version History

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Revision No.** | **Date** | **Description of Change** |
| **Anatoliy Lokshin** | 1.0 | 11/22/2010 | First revision |
| **Maksym Sukhovarov** | 1.0 | 01/14/2011 | Description of VBScript decoder was added. |
|  |  |  |  |
|  |  |  |  |

Functional Design Review

| **Name** | **Title** | **Date Reviewed** | **Date Approved** |
| --- | --- | --- | --- |
| **Kyle Quest** | Architect |  |  |
| **Anatoliy Lokshin** | Development lead |  |  |
| **Julia Kuchmai** | QA Representative |  |  |
|  |  |  |  |

Technical Design Review

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

Engineering Task Worklist Review

| **Name** | **Title** | **Date Reviewed** | **Date Approved** |
| --- | --- | --- | --- |
| **Anatoliy Lokshin** | Development lead |  |  |
|  |  |  |  |
|  |  |  |  |

References

|  |  |  |
| --- | --- | --- |
| **Document Name** | **Author** | **Location** |
| **MetraTech Security Framework Specification** | Kyle Quest | http://seceng.metratech.com/gf/download/docmanfileversion/20/78/MtSecurityFrameworkSpec.doc |
|  |  |  |

Version Configuration

|  |  |  |
| --- | --- | --- |
| **Name** | **Version** | **Additional Comments** |
| MetraNet | 6.05 |  |
|  |  |  |

# System Overview

Reference to the requirement document will be here.

# Assumptions and Dependencies

The decoder subsystem depends on the Configuration loader.

Some kinds of encoding are not unified across different systems, such as HTML entity encoding. Decoders for those encodings work in most intelligent way to cover all widespread presentations.

# Technical Design

## Class diagram and description

A high-level class diagram for the Decoder subsystem is shown on the Figure 1.

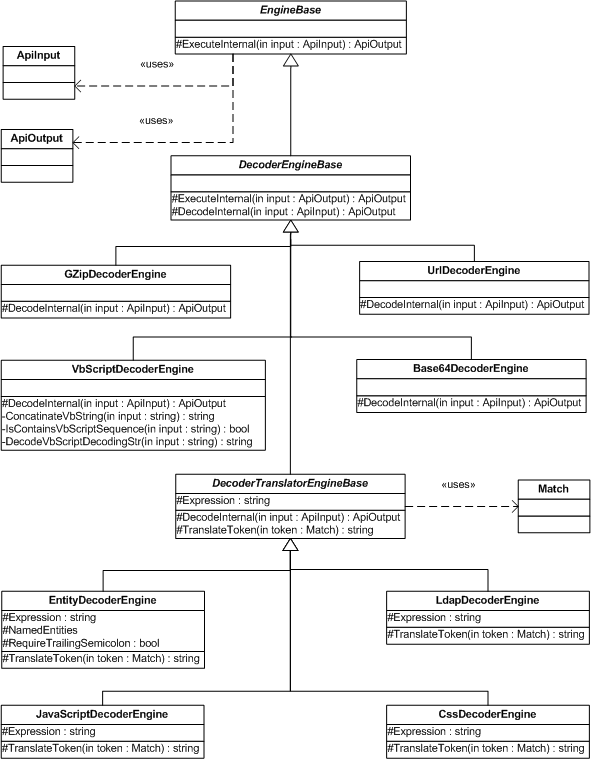


Figure 1. Decoder subsystem class diagram

**DecoderEngineBase** is a common abstract class for all decoder engines. It implements the ExecuteInternal method and defines DecodeInternal abstract method. The ExecuteInternal implementation checks input values for NULLs and empty strings and calls the DecodeInternal method that must contain the decoding logic in descendant classes.

There are 2 kinds of concrete decoder engines. First kind uses the standard decoding techniques. Second kind uses regular expressions to parse the input.

This division is due to .NET framework either has no standard decoding implementation (JavaScript, VB Script, CSS, and LDAP) or the standard implementation does not fit all requirements (HTML, XML).

First kind decoder engines (**UrlDecoderEngine**, **GZipDecoderEngine**, **Base64DecoderEngine**) are inherited directly from the **DecoderEngineBase** abstract class.

URL and GZIP decoders use the standard .NET implementations which fit our requirements. Base 64 decoder provides an intelligent custom implementation of the decoding algorithm to cover deferent encoding variants.

**DecoderTranslatorEngineBase** is a common parent class for all decoder engine those use regular expressions. It implements the DecodeInternal method of the DecoderEngineBase abstract class. Also it defines an abstract property Expression that must return the regular expression text and an abstract method TranslateToken that takes a Match object and translates it to a proper string representation.

**EntityDecoderEngine** is a configurable decoder that allows convert HTML and XML encoded strings. Its configuration specifies a list of named entities and if an escape sequence character must be ended by the semicolon character.

**JavaScriptDecoderEngine**, **LdapDecoderEngine** and **CssDecoderEngine** are very specific and have own implementations.

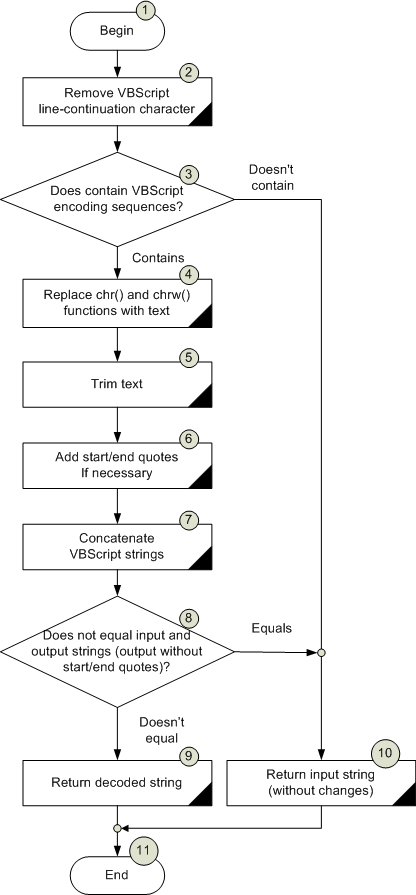
**VbScriptDecoderEngine** is a decoder witch decodes VbScript strings. This decoder's implementation is used into XSS detector. Incorrected input text is not decoded and is transfered transparently, so no exception is thrown.

For example:

1. *<****“VB” & “Script”****>*would be decoded to <***VBScript****>;*
2. *<****“VB” & chr(&h53) & chrw(99) & “Script”****>*would be decoded to <***VBScript****>;*
3. *<****“”””****>*would be decoded to <***”****>;*
4. *<****“””” & “”””*** *>*would be decoded to <***””****>;*
5. *<****“VBSCript” & “ is the “”Visual basic”””*** *>*would be

decoded to < ***VBSCript is the ”Visual basic”****>.*

That is why the decoding algorithm is not a trivial. And consists of many simple functions (see figure 1).



**Figure 1 – “Decoding VBScript encoding string”**

Algorithm has a following steps:

1. Beginning of the algorithm
2. **Remove VBScript line-continuation character** – The ‘**\_\n**’ sequence remove from input string (RemoveLineContinuationChar(in input:string):string function).

For example: input string is ‘*scr\_\nipt*’ after replasing will be ‘*script’*;

1. **Does contain VBScript encoding sequences?** – Сhecks input string by regex for contains ‘**“….” & “….”**’ or ‘**&** **chr(…) &**’ or ‘**&** **chrw(…) &**’ sequences (IsContainsVbScriptSequence(in input:string):bool method);
2. **Replace chr() and chrw() functions with text** . Replace ‘**chr(…)**’ or ‘**chrw(…)**’ with symbol they encode in quotes. ‘**”<symbol>”**’ Also checks for existing ampersands (&) at the start and at the end of substituting functions . It is implemented by Regex (regex name pattern is **RegexPatternGetChrSequence** and regex variable is \_regGetChrw);
3. **Trim text** – Left and right trimming text after replacing functions with text.
4. **Add start/end quotes If necessary** – The result string after convertion checks for existing quote in the start/end of string if does’t exist add to start/end. But if the chr() or chrw() function exists in the start/end string then the quote does not need to add. (AddStartStopQuotesIfNeed(in input:string):string)
5. **Concatenate VBScript strings** – Try to find ‘**”...” & “…”** **& “…”**’ sequence and if it exists concatenate to one string . It is implemented by Regex variable \_concatenateStr.

For example: the input string is “*”VB”&”S”&”c”&”r”&”i”&”p”&”t”*” and

the output string would be ‘*VBScript*’. By the note the start and the end quotes are remove.

1. **Does not equal input and output strings (output without start/end quotes)?** – Checks input and after decoding output strings at the equivalence (IsEqualInputOutputWithoutQuotes(in input:string, in output:string):bool method). By the note outuput string checks without start/stop quotes. It’s necessary because on step 6 the qouters was added. So if strings are not equal go to exit and reurn the decoded string (step #9 in the pecture) step if equal go to exit and return input string without changes (step #10 in the pecture).

## High-Level Design / Tasks

1. Fix an error message and an exception type for URL decoder
2. Implement HTML decoder
3. Implement JavaScript decoder
4. Implement VB script decoder
5. Implement XML decoder
6. Implement LDAP decoder
7. Implement Base 64 decoder
8. Implement GZIP decoder
9. Create user manual for decoder subsystem

# Error list

The following table shows a listing of errors that can occur.

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Code** | **Error Message** | **Description** | **Area** |
| **NullInputDataException** | Input data is null | When input data is null or empty string |  |
|  |  |  |  |

# Outstanding Issues

List all open issues regarding this document.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Date raised** | **Description and Resolution** | **Page/ Section** | **Raised by** | **Allocated to** | **Status** |
| 1 | 1/14/2011 | The VBScript decoder should understand **vbcrlf**, **vbcr**, **vblf** VBScript constants. |  | Maksym  Sukhovarov |  | Open |
|  |  |  |  |  |  |  |