The Code Master

The more you teach, the more you learn.

Which methods should be considered “Sources”, “Sinks” or “Sanitization” ?

One thing I noticed when I first started testing security applications, was that each one, had their own understanding of what is a security vulnerability and which methods should be verified and flagged as vulnerable. I later found out this page (https://www.owasp.org/index.php/Searching_for_Code_in_J2EE/Java) on OWASP. It contains some methods that should be sanitized but I believe it is not the full list.

So, I have created my own list from what I found on other applications, blogs and etc and I was wondering if you (my reader) could help me make this list perfect. If we manage to do this, I think it would be great to update OWASP page. What do you think ?

With this list of methods I performed an evaluation on 5 applications (BlueBlog, PersonaBlog, WebGoat, Roller and Pebble) using 4 Eclipse plug-ins (ASIDE, CodePro, Lapse+ and ESVD(my plug-in)). I got some very promising results. I am really excited.

Do you think there are methods missing or some of these methods should be removed from the list ?

My plug-in is still a prototype but I am receiving some very good feedback. You can see more on:
01 – early-vulnerability-detection-supporting-secure-programming
02 – https://marketplace.eclipse.org/content/early-security-vulnerability-detector-esvd/

My list of “**Sources**“, “**Sinks**” and “**Sanitization**” methods:

Sources:

javax.servlet.ServletRequest

getAttribute
getAttributeNames
getCharacterEncoding
getContentType
getParameter
getParameterNames
getParameterValues
getParameterMap
getProtocol
getScheme
getServerName
getRemoteAddr
getRemoteHost
getLocalName
getLocalAddr
getReader

javax.servlet.http.HttpServletRequest

getAuthType
getHeader
getHeaders
getMethod
getPathInfo
getPathTranslated
getContextPath
getQueryString
getRemoteUser
getRequestedSessionId
getRequestURI
getRequestURL
getServletPath

javax.servlet.http.Cookie

getComment
getDomain
getPath

getName
getValue

javax.servlet.ServletConfig

getInitParameter
getInitParameterNames

javax.servlet.GenericServlet

getInitParameter
getInitParameterNames

java.sql.ResultSet

getString
getObject

java.awt.TextComponent

getSelectedText
getText

java.io.Console

readLine
readPassword

java.io.DataInputStream

readLine
readUTF

java.io.LineNumberReader

readLine

javax.servlet.http.HttpSession

getAttribute
getAttributeNames
getValue
getValueNames

java.lang.System

getProperty
getProperties
getenv

javax.servlet.ServletContext

getResourceAsStream
getRealPath
getHeaderNames

java.util.Properties

getProperty

java.lang.Class

getResource
getResourceAsStream

org.apache.xmlrpc.XmlRpcClient

execute
search

javax.xml.xpath.XPath

evaluate

javax.xml.xpath.XPathExpression

evaluate

Sanitization:

org.owasp.encoder.Encode

forHtml
forHtmlContent
forHtmlAttribute
forHtmlUnquotedAttribute
forCssString
forCssUrl
forUri
forUriComponent
forXml
forXmlContent
forXmlAttribute
forXmlComment
forCDATA
forJava
forJavaScript
forJavaScriptAttribute
forJavaScriptBlock
forJavaScriptSource

java.net.URLEncoder

encode

java.net.URLDecoder

decode

org.apache.commons.lang.StringEscapeUtils

escapeJava
escapeJavaScript
unescapeJava
unescapeJavaScript
escapeHtml
unescapeHtml
escapeXml
unescapeXml
escapeSql
escapeCsv
unescapeCsv

Sinks:

Command Injection

java.lang.Runtime

exec

javax.xml.xpath.XPath

compile

java.lang.Thread

sleep

java.lang.System

load
loadLibrary

org.apache.xmlrpc.XmlRpcClient

XmlRpcClient
execute
executeAsync

Cookie Poisoning

javax.servlet.http.Cookie

Cookie
setComment
setDomain
setPath
setValue

Cross Site Scripting

java.io.PrintWriter

print
println
write

javax.servlet.ServletOutputStream

print
println

javax.servlet.jsp.JspWriter

print
println

javax.servlet.ServletRequest

setAttribute
setCharacterEncoding

javax.servlet.http.HttpServletResponse

sendError
setDateHeader
addDateHeader
setHeader
addHeader
setIntHeader
addIntHeader

javax.servlet.ServletResponse

setCharacterEncoding
setContentType

javax.servlet.http.HttpSession

setAttribute
putValue

HTTP Response Splitting

javax.servlet.http.HttpServletResponse

sendRedirect
getRequestDispatcher

LDAP Injection

javax.naming.directory.InitialDirContext

InitialDirContext
search

javax.naming.directory.SearchControls

setReturningAttributes
connect
search

Log Forging

java.io.PrintStream

print
println

java.util.logging.Logger

config
fine
finer
finest
info
warning
severe
entering
log

org.apache.commons.logging.Log

debug
error
fatal
info
trace
warn

java.io.BufferedWriter

write

javax.servlet.ServletContext

log

javax.servlet.GenericServlet

log

Path Traversal

java.io

File
RandomAccessFile
FileReader
FileInputStream
FileWriter
FileOutputStream

java.lang.Class

getResource
getResourceAsStream

javax.mail.internet.InternetAddress

InternetAddress
parse

Reflection Injection

java.lang.Class

forName
getField
getMethod
getDeclaredField
getDeclaredMethod

Security Misconfiguration

java.sql.DriverManager

getConnection

SQL Injection

java.sql.(Prepared)?Statement

addBatch
execute
executeQuery
executeUpdate

java.sql.Connection

prepareStatement
prepareCall

javax.persistence.EntityManager

createNativeQuery
createQuery

(org|net.sf).hibernate.Session

createSQLQuery
createQuery
find
delete
save
saveOrUpdate
update
load

XPath Injection

javax.xml.xpath.XPath

compile
evaluate

javax.xml.xpath.XPathExpression

evaluate

org.apache.xpath.XPath

XPath

org.apache.commons.jxpath.JXPath

getValue

org.xmldb.api.modules.XPathQueryService

query

org.xmldb.api.modules.XMLResource

setContent

Thank you!
Luciano Sampaio

August 6, 2014 Luciano Sampaio Eclipse, Java, Security

13 thoughts on “Which methods should be considered “Sources”, “Sinks” or “Sanitization” ?”

EclipseUser
September 30, 2014 at 6:50 am

Hi Luciano,

I tried ESVD & its very good. I would like to add one suggestion –
When user click on the security marker, some option related to the vulnerability appears on the hover.
Can we have a message on top stating the vulnerability like – “This element should be sanitized to avoid Cross-Site Scripting(XSS)”. Can we have this title representing the vulnerability for all 11 violations? Right now I guess it is only for Cross Site Scripting.

In Security view, How is the priority numbers defined?

Thanks

Luciano Sampaio
September 30, 2014 at 10:30 am

Hi,

Thank you for downloading and for your suggestions.

01 – My bad!!!! I hardcoded the message "… to avoid Cross-Site Scripting(XSS)", it should be the name of the found vulnerability. I will change that. Thank you!

02 – Because ESVD is still just a prototype, we already predefined all the numbers, they were based on the OWASP Risk Factor Summary (https://www.owasp.org/index.php/Top_10_2013-Details_About_Risk_Factors). We plan to add the option, in which developers can change them as they see fit. However, currently this is not possible.

How do you like the priority numbers ? Do you think they help ?

Thank you!

EclipseUser

October 1, 2014 at 12:57 am

Hi,

Thanks for your quick response 🙂

Priority numbers, if description is available somewhere, it will be helpful. The link you shared is confusing me.I am not able to find what priority no 11 stands for 😕

Few more suggestions from my side to make ESVD more usable –
1) An Explicit option to clear the violation icons from the java files.
2) An option to clear the violations from the Security view
3) Priority is not explained in the document. Detailed explanation of priority numbers & its corresponding violations will definitely be helpful.
4) On all hovers, mentioning about the found violation at the top of the message will be helpful.
5) Export to PDF\Excel option will be a great value add.
6) Instead of setting Run on Save option in Preferences section, if its available on right click of the project, it will be handy & easy.

Thanks!

Luciano Sampaio

October 1, 2014 at 3:57 pm

Hi,

The link I shared was to show how I got the priority numbers. From the OWASP Risk Factor Summary (https://www.owasp.org/index.php/Top_10_2013-Details_About_Risk_Factors) I created my own Risk Factor (http://thecodemaster.net/wp-content/uploads/2014/10/Risk-Factor.png). Take a look and see if it is better to understand now.

01 – What do you mean by “an explicit option” ? Because we do have an option. Check image:
http://thecodemaster.net/wp-content/uploads/2014/10/UI-Dialog-Ignore-Option.png.
02 – I agree, we don’t have this.
03 – I agree.
04 – I agree.
05 – If you right click on the warnings (just one, several or all of them) it should appear a dialog with the option “Copy to clipboard”. If you paste it on the Excel it should be fine. Try it and let me know.
06 – We do not have a “Run” and “Stop” button, this is true, but we do have an “Enable” and “Disable” button, which is almost the same, don’t you agree ? 🙂

Talk to you soon.

EclipseUser

October 1, 2014 at 12:59 am

I would like to know when is the next release of the plugin & what are the proposed enhancements\changes?

Luciano Sampaio

October 1, 2014 at 3:42 pm

The plug-in is just a proof of concept, until I finish my thesis, I will not have time to work on it. I did find several places where I can improve it in order to make ESVD better. However, that will have to wait a little bit.

Thanks.

EclipseUser

October 6, 2014 at 2:40 am

Hi,
Thanks for your elaborate explanation on my queries on ESVD. Your response is helpful.

Hope to see ESVD updated version soon 🙂

EclipseUser

October 10, 2014 at 2:41 am

Hi,

Just curious to know whether you have uploaded the source code in git\svn?

Luciano Sampaio

October 10, 2014 at 8:54 am

Hi,

So far the source code is only available to the members of my research group.

Thanks!

EclipseUser

October 15, 2014 at 1:40 am

Hi Luciano,
Can you give me the list of reference materials you went through for creating this plugin ? I would also like to know if any parser has been used for finding the security violation.

Thanks.

Luciano Sampaio

October 15, 2014 at 9:23 am

Hi,

The main references were these 4.
01 – https://wiki.eclipse.org/Eclipse_Corner
02 – http://www.vogella.com/tutorials/eclipse.html
03 – Eclipse 4 Plug-in Development by Example
04 – Eclipse Plug-ins Third Edition Dec 2008

I created my own parser.

Thanks and let me know if there is anything else I can help you with.

Pingback: Java Plug-in that checks vulnerability state (Featured Guest) | ODS3 Cyber Security Academy

---

Robert

February 15, 2016 at 11:58 am

All sources and sinks provide the ability to open a new stream for reading or writing. By default, other operations are all implemented by calling one of these methods to get a stream, doing something, and then ensuring that the stream is closed.
