

# **Configuration Manual**

MSc Internship MSc in Cyber Security

Lyubka Dencheva Student ID: x20195907

School of Computing National College of Ireland

Supervisor:

Mark Monaghan

#### National College of Ireland

#### **MSc Project Submission Sheet**



**School of Computing** 

Student Name:	Lyubka Dencheva	
Student ID:	x20195907	
Programme:	MSc in Cyber Security	<b>Year:</b> 2022
Module:	Internship	
Supervisor:	Mark Monaghan	
Submission Due Date:	15/08/2022	
Project Title:	Comparative analysis of Static ap (SAST) and Dynamic application using open-source web application	oplication security testing security testing (DAST) by on penetration testing tools

#### Word Count: 1391

#### Page Count: 11

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

<u>ALL</u> internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Lyubka Dencheva

Date: 15th August, 2022

#### PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple	
copies)	
Attach a Moodle submission receipt of the online project	
submission, to each project (including multiple copies).	
You must ensure that you retain a HARD COPY of the project, both	
for your own reference and in case a project is lost or mislaid. It is not	
sufficient to keep a copy on computer.	

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

#### **Office Use Only**

Office use Offiy	
Signature:	
Date:	
Penalty Applied (if applicable):	

## **Configuration Manual**

Lyubka Dencheva Student ID: x20195907

### **1** Introduction

The configuration manual describes the methodology and execution framework presented in the master's thesis, providing information on the installation, configuration, and execution of the necessary tools for the purposes of a research paper.

System configuration begins with the installation and configuration of a Kali Linux Virtual Machine to provide high functionality and flexibility to perform the intended tests. After its successful launch, the installation and configuration of the used Static Application Security Testing (SAST) and Dynamic Application Security Testing (DAST) tools is performed, followed by the actual execution of the security vulnerability scan and the generation of its results. Subsequently, they are used by the selected Benchmark to prepare scorecards with summarized, systematized, and accurate results. Based on them, the comparative analysis between open-source SAST and DAST tools is performed, which is the focus of the master's thesis.

### 2 System configuration

Host machine
 Operation System (OS): Microsoft Windows 10 Enterprise
 System type: x64 – based PC
 Processor: Intel(R) Core(TM) i5-4210M CPU @2.6GHz, 2601Mhz, 2 Core(s), 4 Logical
 Processor(s)
 Memory: 12GB

2) Virtual machine
Operation System (OS): Kali Linux version 5.9
System type: 64-bit operating system
Processor: 4 processor cores
Memory: 7451 MB
Video memory: 16 MB

#### 2.1 Installation and configuration of the Virtual Machine Kali Linux

In order to install and use a Virtual Machine, a platform is needed to run it. For this purpose, the open source software Oracle VM VirtualBox Manager version 6.1 is used, downloaded from its official site<sup>1</sup>, which is installed on the Windows OS host machine and the Operation System Kali Linux is configured on it. Regarding this configuration, Jon Watson's research

<sup>&</sup>lt;sup>1</sup>Oracle VM VirtualBox: https://www.virtualbox.org/

paper<sup>[1]</sup> explains in great detail, step by step, how the entire process is carried out. The end result is to start the virtual machine as shown in Figure 1.



Figure 1: Virtual Machine Kali Linux on Oracle VM VirtualBox Manager

## 3 Environment Setup

For the purposes of the master's thesis, in addition to a Kali Linux virtual machine, Static Application Security Testing (SAST) and Dynamic Application Security Testing (DAST) tools are needed, the productivity and efficiency of which are being investigated. Based on them and a specially selected Benchmark, a qualitative, detailed and accurate comparative analysis is performed between the these two types of security testing. The overall process to achieve it through specialized, installations and configuration is presented in the following subsections.

### 3.1 Installation and configuration of Benchmark

The implementation process begins with downloading the open-source OWASP Benchmark zip files from the GitHub platform<sup>2</sup> and placing it on the Virtual Machine. Next, unzipping the file with the unzip command and accessing the VMs directory, via the following command:

cd root/Desktop/BenchmarkJava-master/VMs

Then, it is necessary to build and run the docker, which is done by successively running the files BuildDockerImage.sh and runDockerImage.sh with the bash command (see Figure 2).

<sup>&</sup>lt;sup>2</sup> OWASP Benchmark Java Project: https://github.com/OWASP-Benchmark/BenchmarkJava



**Figure 2: Running Docker scripts** 

After the above commands are executed, OWASP Benchmark is started and can be accessed via the web browser at the localhost URL on port 8443 as follows (see Figure 3): https://localhost:8443/benchmark

At the above URL, a benchmark has been positioned and loaded and it is now ready to start scanning for security vulnerabilities, as shown in Figure 3:



Figure 3: OWASP Benchmark Java Project

### **3.2 Installation and configuration of SAST tools**

### **3.2.1 LGTM.COM**

LGTM.COM is an online tool that is a code analysis platform to identify vulnerabilities and prevent them from reaching production. It requires no installation and is accessed through the web browser on its official website<sup>3</sup>. In this tool, there is an extremely large number of already loaded open-source projects, among which is OWASP Benchmark (see Figure 4).

<sup>&</sup>lt;sup>3</sup> LGTM.COM: https://lgtm.com/

Thousands of projects already on LGTM		
With over 39 million commits by more than 700,000 developers analyzed for 135,821 open source projects (and counting!), there's a good chance we're already analyzing your projects!	neo4j/neo4j	apache/struts Java D JavaScript C
Start exploring results by searching for specific projects and contributors below.	230 354.1k 105 Contributors Lines of code Alerts	80 122.2k 190 Contributors Lines of code Alerts
Or log in to see prioritized alert results for your projects we're already analyzing, or to add more projects to LGTM.	meteor/meteor JavaScript C Python A+	angular/angular JavaScript
OWASP BenchmarkJa	379 149k 380 Contributors Lines of code Alerts	1.6k 168.2k 265 Contributors Lines of code Alerts
OWASP-Benchmark/BenchmarkJava View project >		

#### Figure 4: LGTM.COM

### 3.2.2 Visual Code Grepper (VCG)

The open-source Visual Code Grepper (VCG) scanning tool is installed on the Windows host machine for faster and more convenient scanning. The installation files of the tool are downloaded from the GitHub platform<sup>4</sup> and the installer is launched to begin the installation. Once completed, VCG is launched by clicking on the icon that is generated after successful installation.

#### **3.3 Installation and configuration DAST tools**

#### 3.3.1 Arachni

The files provided on the official site<sup>5</sup> are used to install the Arachni open-source tool. After downloading, they are positioned on the Linux machine, with the aim of higher speed and flexibility. Then, from the Linux terminal, set the tool directory and specifically the bin directory as follows:

cd root/Desktop/Arachni/bin

After that, the command to start arachni\_web is set (see Figure 5), after which the tool is ready to perform security vulnerability scans. It is accessed through the web browser as the URL of the localhost with port 9292 (see Figure 6):

http://localhost:9292

Then, it is necessary to enter the basic data for access which are:

Email – admin@admin.admin Password – administrator

<sup>&</sup>lt;sup>4</sup> Visual Code Grepper (VCG): https://github.com/nccgroup/VCG

<sup>&</sup>lt;sup>5</sup> Arachni: https://www.arachni-scanner.com/download/



Figure 5: Starting of arachni\_web

🐼 OWASP Benchmark 🗙 🗙	🚱 Arachni		+					
$\leftarrow \rightarrow C$ (i) localhost:9292/				o <del>.</del>	<	☆		
Arachni v1.6.1.3 - WebUI v	/0.6.1.1						Sign	
	Arachni is heading towards	ls obsole	lescence, try out its next-gen successor Ecsypno SCNR!					
Sign in								
Email								
admin@admin.admin								
Password								
•••••								
Remember me								
Sign in								

Figure 6: Aranchi

After the successful entry into the system, the admin panel of the Arachni is visualized, from where the configuration and management of the tool is carried out.

#### 3.3.2 OWASP ZAP

The installation files for the open-source tool OWASP ZAP can be downloaded from the official website<sup>6</sup>. Then, they are loaded through the Linux terminal by running the zap.sh file as shown in Figure 7. This launches OWASP ZAP and the tool is ready to perform a security vulnerability scan.

<sup>&</sup>lt;sup>6</sup> OWASP ZAP: https://www.zaproxy.org/docs/desktop/releases/2.9.0/



Figure 7: OWASP ZAP

## 4 Implementation and execution of used tools

#### 4.1 SAST tools

#### **4.1.1 LGTM.COM**

As already established, the LGTM.COM tool contains an already analyzed OWASP Benchmark Project. Therefore, to obtain the results of the performed security vulnerability scan, it is only necessary to download the files from the scan, as shown in Figure 8:

Active alerts 3f11b4a 🛦 (	•		
Alert filters			Export alerts 🛓
Severity 2 × Query ×	Tag 1 × Language ×	Group by query	Export alerts in SARIF format. Only alerts matching the current filters will be exported.

Figure 8: LTGM.COM – export results

### 4.1.2 Visual Code Grepper (VCG)

There is an important feature when performing a security vulnerability scan with Visual Code Grepper (VCG). As soon as the tool is launched, it is necessary to select the programming language in which the scan will be performed. The test files and/or directories are then loaded, and the vulnerability scan is started, as shown in Figure 9:

File Edit View S C:\Users\Lyubka\Desktor	can Settings He Full Scan	lp		<u>.</u>		
C:\Users\Lyubka\Desktor	Full Scan					
Target Files Results S	Visual Code/Com Scan Code Only (	nment Breakdown Ignore Comments)				
C:\Users\Lyubka\Deskt C:\Users\Lyubka\Deskt C:\Users\Lyubka\Deskt C:\Users\Lyubka\Deskt	Scan For Bad Fun Scan Comments	ctions Only (As Defined in Config Only	File)	Benchmark Test 00001 java Benchmark Test 00002 java Benchmark Test 00003 java Benchmark Test 00004 java		
C:\Users\Lyubka\Deskt	Show All 'FixMe'	Show All 'FixMe' Comments				
C:\Users\Lyubka\Deskt C:\Users\Lyubka\Deskt C:\Users\Lyubka\Deskt C:\Users\Lyubka\Deskt	Sort Rich Text Results on Severity Sort Rich Text Results on FileName			Benchmark Test00007 java Benchmark Test00008 java Benchmark Test00009 java		
C:\Users\Lyubka\Deskt C:\Users\Lyubka\Deskt C:\Users\Lyubka\Deskt	Filter Results			Benchmark Test00011.jav Benchmark Test00011.jav		
C:\Users\Lyubka\Deskt C:\Users\Lyubka\Deskt	Delete Selected It	em(s)	Del	Benchmark Test00013 java Benchmark Test00014 java		
C:\Users\Lyubka\Desktop C:\Users\Lyubka\Desktop C:\Users\Lyubka\Desktop C:\Users\Lyubka\Desktop C:\Users\Lyubka\Desktop C:\Users\Lyubka\Desktop	BenchmarkJava-master BenchmarkJava-master BenchmarkJava-master BenchmarkJava-master BenchmarkJava-master BenchmarkJava-master	\src \main \java \org \owasp \benchmar \src \main \java \org \owasp \benchmar	k \testcode \ k \testcode \ k \testcode \ k \testcode \ k \testcode \ k \testcode \	Benchmark Test00015 java Benchmark Test00016 java Benchmark Test00017 java Benchmark Test00018 java Benchmark Test00019 java Benchmark Test00020 java		

Figure 9: VCG Scan

After the scan is complete, the results can be exported in various formats, but for the current research work, xml format is used (see Figure 10).

File	Edit	View	Scan	Settings	Help		
New Target Directory Ctrl+N New Target File Ctrl+T				Ctrl+N	src\main\java\org\owasp\benchmark\testcode		
				Ctrl+T			
	Save Re	esults as	Text	Ctrl+S			
	Clear				on Primitive Data Type		
	Export	Results a	s XML		hmarkJava-master/src/main/java/org/owasp/benchmark/te		
Import Results from XML File			from XM	L File	assName.lastIndexOf('.') + 1 + "BenchmarkTest".lengt		
	Export Results to CSV File Import Results from CSV File		e	t Validation chmarkJava-master\src\main\java\org\owasp\benchmark\ tained in the HttpServletRequest without validation or sanitis			
			/ File				
	Export	Code Me	etadata a	s XML	<pre>kie[] cookies = request.getCookies();</pre>		
	Exit			Alt+F4	on Primitive Data Type		
Line	. 0J - C.	(Users (L	упрка		enchmarkJava-master\src\main\java\org\owasp\benchmark\te		
The	code a	opears t	o be co for (in	mying out it i = 0;	a mathematical operation on a primitive data type. In some circun !foundUser && i < cookies.length; i++) {		
HIG	l: Poten	tially Un	safe Co	de - Poor I	nput Validation		
Line	83 - C:	Users\L	yubka\	Desktop\B	enchmarkJava-master\src\main\java\org\owasp\benchmark\te		
The oreme	applica mberMe	tion app .setPa	bears to th(requ	use data ( lest.getRe	contained in the HttpServletRequest without validation or sanitisat equestURI());		
POT		SSUE De	In a Kally	Illes etc. C.	de Public Class Net Declased as Final		

Figure 10: VCG – export results

#### 4.2 DAST tools

#### 4.2.1 Arachni

The start of the scan is set from the Arachni admin panel, which is accessible through the web browser, as already discussed. In the target field, the URL address of the OWASP Benchmark is placed and the scan is started, as shown in Figure 11:

Arachni is heading towards obsolescence, try out its ne	kt-gen successor Ecsypno SCNR!
Start a scan The only thing you need to do is provide some basic information and make	a simple choice about the type of scan you want to perform.
https://192.168.1.232:8443/benchmark/	Default (Global)
Full URL of the targeted web application (must include the appropriate protocol, http or https).	Configuration profile to use.
Description	Share with:
	Regular User
You can use Markdown for text formatting.	
Advanced options	
Go!	

#### Figure 11: Arachni Scan

After completing the scan for security vulnerabilities, the tool presents an option to export the obtained results. They are used to perform further analysis (see Figure 12).

TOGGLE VISIBILITY OF	https://192.168.1.232:8443/benchmark/			
ACTIONS	Edit description			
C Share				
<ul> <li>Full edit</li> <li>Download report as:</li> </ul>	• This scan has the logged the following errors (if something looks like a bug feel free to report it):			
HTML JSCN Exports the audit results as an XML (xml) file. XML YAML AFR	[2022-08-02 02:16:00 +0100] /home/lyubka/Desktop/arachni-1.6.1.3-0.6.1.1/.syste [2022-08-02 02:16:00 +0100] /home/lyubka/Desktop/arachni-1.6.1.3-0.6.1.1/.syste			
	<pre>[2022-08-02 02:16:00 +0100] /home/lyubka/Desktop/arachni-1.6.1.3-0.6.1.1/.syste [2022-08-02 02:16:00 +0100] /home/lyubka/Desktop/arachni-1.6.1.3-0.6.1.1/.syste [2022-08-02 02:16:00 +0100] /home/lyubka/Desktop/arachni-1.6.1.3-0.6.1.1/.syste</pre>			
	✓ The scan completed in 18:44:54.			

Figure 12: Arachni – export results

#### 4.2.1 OWASP ZAP

Before starting the scan in OWASP ZAP, it is good to configure the scan setting by increasing the number of possible results (see Figure 13). Then, in the field for URL attack address, the address of OWASP Benchmark is set and the scan for security vulnerabilities is

started. This tool, as well as the ones discussed above, also provides an opportunity to export the results (see Figure 14) that will be needed for the research work.

	Options	×
् 🗶	Active Scan	0
Active Scan P Active Scan Input Vector AJAX Spider Alerts Anti-CSRF Tokens API Applications Breakpoints Callback Address Check For Updates Client Certificate Connection Database Dienlax	Number of Hosts Scanned Concurrently:         I       I	
Dynamic SSL Certificates Encode/Decode Extensions Forced Browse Fuzzer	<ul> <li>Inject plugin ID in header for all active scan requests.</li> <li>Handle anti-CSRF tokens (experimental functionality).</li> </ul>	
Global Alert Filters Global Exclude URL HTTP Sessions HUD	<ul> <li>✓ In Attack Mode prompt to rescan nodes when scope changed.</li> <li>✓ In Attack Mode always rescan nodes when scope changed.</li> </ul>	
JVM Keyboard Language	Attack Mode Scan Policy:	Ť
Reset to Factory Defaults	Cancel	к

Figure 13: OWASP ZAP – configuration

<u>F</u> ile <u>E</u> dit ⊻iew <u>A</u> nalyse	<u>Report</u> <u>T</u> ools <u>I</u> mport <u>O</u> nline <u>H</u> elp			
Eile Edit View Analyse Standard Mode    Standard Mode   Contexts  Default Context  Solution  Sites	Report         Tools         Import         Online         Help           Generate         HTML Report         Generate         McReport         Generate         SON Report         Generate         SON Report         Export         McReport         Export         Son Report         Export Response(s) to File         Export Response(s) to File         Export All URLs to File         Export Selected URLs to File         Export URLs for Context(s)         Compare with Another Session		<ul> <li>▶ </li> <li>♥ Quick Start ♥ → Requine</li> <li>♥ Please be aware that you share</li> <li>♥ URL to attack:</li> <li>♥ Progress:</li> </ul>	est Response ( est Response ( unch an automated scan against an appli nould only attack applications that you hav https://localhost:8443/benchmark/ ( with Firefox Headless ( Attack Stop Using traditional spider to discover the co
History 🔍 Search	Pu Alerts 📄 Output 👌 Active S	Scan 🔆 Spider 🖉 🛎 🕂		
i ≫ New Scan ≣ Progress:	0: https://localh:8443/benchmark		51%	Surrent 🖤 Current

Figure 14: OWASP ZAP – export results

## **5.** Generate a scorecard for analysis

After completing the scan for security vulnerabilities with the above-mentioned and discussed SAST and DAST tools, the results are exported as xml and serif formats, as they are accepted by selected Benchmark for further analysis. These files are placed in the OWASP Benchmark directory, in folder results, as shown in Figure 15:



Figure 15: OWASP Benchmark - Results folder

Then the createScorecard.sh script is run from the Linux terminal (see Figure 16).

(root & kali)-[/home/lyubka/Desktop/BenchmarkJava-test]
<pre>Up bass createscorecards.sn [INFO] BuildTimeEventSpy is registered. [INFO] Scanning for projects</pre>
[INFO] DEVICES
[INFO] org.owasp:benchmark ≻
[INFO][ war ]
[INF0] Content Benchmark_V1.2_So
default YAML Scoring config file found and loaded. File used was:
jar:file:/root/.m2/repository/org/owasp/benchmarkutils-maven-plugin/1.3/benchmarkutils-maven- plugin-1.3.jar!/defaultscoringconfig.yaml
Deleting previously generated scorecard files in: /home/lyubka/Desktop/BenchmarkJava-test/sco
Read expected results from file: /home/lyubka/Desktop/BenchmarkJava-test/expectedresults-1.2. csv
2740 results found.
Analyzing results from ResultsLGTM.sarif Actual results file generated: /home/lyubka/Desktop/BenchmarkJava-test/scorecard/Benchmark_v1 .2_Scorecard_for_LGTM_v1.31.0-SNAPSHOT.csv
Analyzing results from ResultsArachni.xml Actual results file generated: /home/lyubka/Desktop/BenchmarkJava-test/scorecard/Benchmark_v1 .2_Scorecard_for_Arachni_v1.6.1.3.csv
Analyzing results from ResultsZAP.xml
WARNING: ZAP CWE not mapped to expected test suite CWE: 120 Actual results file generated: /home/lyubka/Desktop/BenchmarkJava-test/scorecard/Benchmark_v1 .2_Scorecard_for_OWASP_ZAP_v2.9.0.csv
Analyzing results from ResultsVCG.xml
Actual results file generated: /home/lyubka/Desktop/BenchmarkJava-test/scorecard/Benchmark_v1 .2_Scorecard_for_VisualCodeGrepper.csv Tool scorecards computed.
Vulnerability scorecards computed.
scorecard written to: /nome/lyubka/Desktop/BenchmarkJava-test/scorecard/Benchmark_v1.2_Scorec ard for Arachni v1.6.1.3.html
Scorecard written to: /home/lyubka/Desktop/BenchmarkJava-test/scorecard/Benchmark_v1.2_Scorec ard for LGTM v1.31.0-SNAPSHOT.html
Scorecard written to: /home/lyubka/Desktop/BenchmarkJava-test/scorecard/Benchmark_v1.2_Scorec
ard_of_services.come/lyubka/Desktop/BenchmarkJava-test/scorecard/Benchmark_v1.2_Scorec ard_for_VisualCodeGrepper.html
Benchmark scorecards complete.

Figure 16: Script createScorecard.sh

This script generates scorecards for each of the Static Application Security Testing (SAST) and Dynamic Application Security Testing (DAST) tools and placed generated files in the OWASP Benchmark directory in the scorecard folder, as shown in Figure 17:



Figure 17: scorecard folder

The generated scorecards are extremely efficient and useful for performing a comparative analysis between the investigated types of security testing tools, SAST and DAST, which are the subject of this research work.

### References

[1] Watson, J. (2008). VirtualBox: Bits and Bytes Masquerading as Machines. *Belltown Media*, 2008(166), 1.