

# **Configuration Manual**

MSc Research Project

Cyber Security

# Noorullah Mohammed Sakhib

Student ID: X21225273

School of Computing

National College of Ireland

Supervisor: Niall Heffernan

#### National College of Ireland



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

#### **School of Computing**

- Student Noorullah Mohammed Sakhib Name:
- **Student ID:** X21225273
- **Programme:** MSc in Cybersecurity

**Year:** 2022-2023

- Module: MSc Internship
- **Lecturer:** Niall Heffernan

Submission Due Date:

- 14/07/2023
- ProjectPROTECTING USER DATA STORED LOCALLY ONTitle:SMARTWATCHES USING ASCON ENCRYPTION: ALIGHTWEIGHT CRYPTOGRAPHY APPROACH

Word		
Count:	906	Page Count: 7

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: Noorullah Mohammed Sakhib

**Date:** 14/07/2023

### 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	use Only						
Signature:							
Date:							
Penalty Applied (if applicable):							

# **Configuration Manual**

Mohammed Sakhib Noorullah

Student ID: x21225273

#### 1. Introduction

The configuration document offers comprehensive details regarding the hardware, software, and implements used for the research project's implementation, testing, and training. Additionally, it gives specifics on how to execute the program's code and defines the implementation process and how results are obtained. The project consists of technique to encrypt and decrypt the data stored on a smartwatch emulator and analyse the obtained results.

#### 2. Environment Setup

#### 2.1 Configuration Setup

To reproduce this study, the pre-requisites indicated below has been used, which entail various installations as listed below, are necessary.

The experiment can be performed on an isolated virtual environment of a stand-alone machine. The use of a virtual lab environment is not compulsory; hence we have performed this experiment on the local machine and the details can be seen from Table 1.

1.	Table: Host System Configuration

Host System configuration				
<b>Operating System type</b>	Windows 10 (64-bit)			
Memory	16 GB RAM			
Processor	Intel(R) Core(TM) i7-4510U CPU @ 2.00GHz			
GPU	NVIDIA GeForce 840M (2 GB DDR3 dedicated)			

### 3. Installation of Software and tools

For this research we require an integrated development environment (IDE) which we form a base for all our operation that needs to be performed. Android studio is an official ID for Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. For this research we will utilize this application to perform our analysis on smartwatch encryption. The programming languages used in this application are mainly Kotlin and Java.

The Android studio version and the programming languages used in this application are mentioned in Fig 1 and Table 2.



#### Fig 1: Android studio version

#### 2. Table: Lab Configuration

Lab Configuration							
Android Studio version	AndroidAndroid Studio Flamingo   2022.2.1 Patch 2Androidhttps://androidstudio.googleblog.com/2023/05/android-studio-flamingo-patch-2Studio versionnow.html						
Programming languages	Kotlin and Java						

#### 3.1 Extraction and Environment setup

The steps to extract code from a GitHub repository and run it in Android Studio as follows.

- *a. Clone the repository:*
- Open a web browser and go to the GitHub repository https://github.com/Noor17892/Ascon\_Encryption\_Project
- Click on the "Code" button and select "Clone" to copy the repository URL.
- b. Launch Android Studio:
- Launch Android Studio on your machine.
- c. Open Project from Version Control:
- In Android Studio, select "Check out project from Version Control" on the welcome screen or from the "File" menu.
- Choose "Git" from the dropdown list.
- d. Paste Repository URL:
- Paste the repository URL you copied from GitHub into the "URL" field.
- Choose a directory where you want to save the project on your local machine.
- Click "Clone" to download the repository.
- e. Build and Sync Project:
- Android Studio will open the project and begin syncing Gradle.
- Wait for the Gradle build to complete. This might take a moment as Android Studio fetches dependencies.

- f. Run the App:
- Once the project is synced, open the main app module (named "app", "common", "wearable").
- Click on the green "Run" button (a play icon) to build and run the app on an emulator or connected device.

**Note**: If there is any error while importing the code from GitHub repository then extract the .rar files from the zip folder (Ascon Encryption Project code) and paste all the files in the Android Studio folder.

#### 4. Execution

Once the setup is complete and Gradle build is done, we can start with the replication of our experiment. The Android studio comprises of 3 main modules.

a. **app**: The app module has been designed to perform operation of an android emulator environment which is illustrated in Fig 2. The MainActivity.kt comprises the code which shows the output on Android emulator as illustrated in Fig 3, that can be selected from device manager.

The file to be analysed is first added into the common module and then file name is added into the MainActivity.kt of this module and the run is initiated. For this experiment we have used a test file "projecttest" which is a .txt format and which contains data in text format, that would help in understanding on how the application is performing encryption and decryption.



Fig 2: App module for smartphone emulator



Fig 3: Android emulator

b. **Common**: The common module has been designed stored the Ascon encryption code and the data set is stored which will be processed for performing encryption and decryption operation and the performance is calculated as illustrated in Fig 4.

S	tudioProjects $ angle$ common $ angle$ src $ angle$ main $ angle$ res $ angle$ raw $ angle$ 🖞	projecttest.txt	t 🔨 🔺 All Tests 🔻 🖍 Wear OS L
dle	🔺 Android 👻 😯 😇 🛨 💠 🕆	— 🕌 Main/	Activity.kt 🛛 🚦 projecttest.txt 🗡
Gra	➤ T <sub>il</sub> common		this is for testing
R	> 🖿 manifests		
L.	🗡 🖿 java		
nage	Image: Common Commo		You are the Chief Information Security Of
e Mai	> 🗖 utils		The board is extremely nervous about your
urce	G Ascon128av12		Your project is to put together a report
Reso	AsconHelper		1. Discuss one specific threat – What is
÷	> Image: com.study.common (androidTest)		2. Discuss four recent high-profile atta
	<ul> <li>com.study.common (test)</li> </ul>		3. Discuss three security controls (one
'ojec	C ExampleUnitTest		The security controls that we would like
4 -	🌠 java (generated)		Operational: (principle of least privileg
	Y 📭 res		Providing limited information to the empl
nmit	✓ Image: Y manual control of the second		
Con	🖨 flashall.bat		
÷	projecttest.txt		
	roboto_mono.zip		
ests	spwquestions.docx		
Requ	res (generated)		

Fig 4: Common Module

c. Wearable: The wearable module has been designed to perform operation of a smartwatch emulator environment which is illustrated in Fig 5. The MainActivity.kt comprises the code which shows the output on smartwatch emulator as illustrated in Fig 6 that can be selected from device manager. The file to be analysed is first added into the common module and then file name is added into the MainActivity.kt of this module and the run is initiated. For this experiment we have used a test file "projecttest" which is a .txt format and which contains data in text format, that would help in understanding on how the application is performing encryption and decryption.

$_{ m by} angle$	wearable $\rangle$ presentation $\rangle$ 🚏 MainActivity.kt $\rangle$ 🍞 MainAct	tivity $\rangle$	operform 💿	Encryption 🔨	🗌 🔛 All Tests 🔻 📃 🕼 Wear OS Large Round API 30 💌 ] 🕨 🇯 🕼 🔊 👻 🚓 🔲 🛛 Git:
dle	🛎 Android 👻 😌 🗄 🐳 —	i N	lainActivity	.kt 🛛 🚦 projec	ttest.txt ×
Gra	> 📑 арр				
R	> 🖬 common			nnivate fun	nerformEncryntion() {
_	🗡 🖿 wearable				Jaunch J this Country (
age	> 🖿 manifests			Toscohe	(Hencefore the Sile II)
Man	Y 🖿 java			Lõg	("opening the file")
Ce	✓ ► com.study.wearable.presentation			res	ources.openRawResource(R.raw. <u>projecttest</u> ). <b>use</b> {    it:InputStream
nose	> 🖿 theme				log("reading the file")
~ •	🙀 MainActivity.kt				<pre>val bytes = it.readBytes()</pre>
	> 🖿 res				
t	res (generated)				
roje	> 🗬 Gradle Scripts				<pre>log("original file: size=\${bytes.size} data=\${bytes.toString(Stan</pre>
					log("encryption_started")
mit					val startTime = System currentTimeMillis()
Cen					log("key: size-\$ key size} data-\$ key tothing(StandardChansets )
÷					log( new, size-\$(new, size) data-\$(new, tostring(standardenarbers.)
					tog( nonce. size-\$inonce.size) uata=\$inonce.tostring(stanuaruchar
lests					Log(
tequ					"additionalData: size= <b>\${additionalData.size}</b> data= <b>\${</b>
- <u>-</u>					additionalData.toString(

Fig 5: wearable Module



Fig 6: Smartwatch emulator

Once the module is selected and run successfully, the emulator can be initiated from the device manager as shown in Fig 7 and Fig 8. The button for encryption selected and the output can be observed in logcat and the performance can be seen in profiler tab respectively, which is present at the bottom of the android simulator application screen.

Lo	gcat: Logcat						\$	
	- Wear OS Lar	ge Round API 30 (emulat	tor-5556) Android		<b>T</b> * package:mine tag:logger			
	2023-08- 2023-08- 2023-08- 2023-08- 2023-08- 2023-08-	04 10:23:23.321 04 10:23:23.321 04 10:23:23.321 04 10:23:23.321 04 10:23:23.536	1573-4105 1573-4105 1573-4105 1573-4105 1573-4105	Logger Logger Logger Logger Logger	com.study.wearable com.study.wearable com.study.wearable com.study.wearable com.study.wearable	0 0 0 0	encryption started key: size=10 data=12345607890123456 nonce: size=10 data=6543210987654321 additionalData: size=20 data=Ascon Encryption cipherData: size=2323 b data=!ZH0+euj0+0+0+0+0+0+0+0+0+0+0+0+0+0+0+0+0+0+0+	f <b>●</b> "●
	2023-08 2023-08 2023-08	.04 10:23:23.536 .04 10:23:23.536 .04 10:23:23.537	1573-4105 1573-4105 1573-4105	Logger Logger Logger	com.study.wearable com.study.wearable com.study.wearable	D D D	<pre>GGI66067600\&amp;&gt; 0m -0X1000100. E0-00-0;0+1°m00D66(-000-0 0!W@=J0-0-D?-0-00007m-0x0g2310200-050,0t6u=0L;}000_050 b000/:E020j00*0+8000j-000C00002-0Z0000 encryption finished encryption time: 215 ms</pre>	•]•G• 7
ų	Git 🕨 Run	🕫 Profiler 🖃 Log	cat 🛛 🕀 App Qu	ality Insights	≺ Build ≔ TODO	🖸 Services 🛛 💂 App Ir	nspection 🗖 Layout Inspe	ctor





Fig 8: Profiler output

## 5. References

- https://androidstudio.googleblog.com/2023/05/android-studio-flamingo-patch-2-now.html
- <u>https://github.com/Noor17892/Ascon\_Encryption\_Project</u>