# Investigación a cerca de cómo cifrar y

# analizar paquetes de datos

Por: Andrés Felipe Diaz G.

Proyecto aplicado en TIC[[1]](#footnote-1)

Universidad Pontificia Bolivariana

Medellín

2020

# Índice

[Investigación a cerca de cómo cifrar y 1](#_Toc38999833)

[analizar paquetes de datos 1](#_Toc38999834)

[Índice 2](#_Toc38999835)

[Justificación 4](#_Toc38999836)

[Objetivos 5](#_Toc38999837)

[ Objetivo general 5](#_Toc38999838)

[ Objetivos específicos 5](#_Toc38999839)

[MobSF, primer acercamiento 6](#_Toc38999840)

[Instalación 6](#_Toc38999841)

[Para Windows 7](#_Toc38999842)

[Para Linux/Mac 8](#_Toc38999843)

[Ejecutar MobSF 10](#_Toc38999844)

[Funcionamiento 11](#_Toc38999845)

[Análisis estático 11](#_Toc38999846)

[Análisis dinámico 11](#_Toc38999847)

[Ventajas y Desventajas 13](#_Toc38999848)

[Ventajas 13](#_Toc38999849)

[Desventajas 13](#_Toc38999850)

[Posibles soluciones 14](#_Toc38999851)

[Workbox Google Analytics 14](#_Toc38999852)

[Conclusiones 15](#_Toc38999853)

[Cibergrafía 16](#_Toc38999854)

# Justificación

Esta investigación se llevó a cabo como requerimiento para la continuación del proyecto “EduApps”[[2]](#footnote-2), más específicamente a la hora de desarrollar la aplicación de “Estudiantes” la cual requiere garantizar la integridad de los datos y el análisis correspondiente de estos con el fin de entregarle a los usuarios un entorno seguro en el cual realizar las actividades requeridas.

# Objetivos

## Objetivo general

Dar a conocer el entorno de MobSF, la instalación de este framework y otras alternativas que se pueden implementar para suplir las necesidades en el desarrollo.

## Objetivos específicos

1. Detallar rigurosamente la forma en la que se implementa MobSF, en qué entorno se puede implementar y las funciones que esto conlleva.
2. Responder a la pregunta ¿Puede este implementarse sin necesidad de tener conexión a internet?
3. Buscar alternativas (otros framworks[[3]](#footnote-3)) que supla con las necesidades de implementación y protección de datos.

# MobSF, primer acercamiento

El primer acercamiento a este tema fue la investigación de este framework con el fin de entender el funcionamiento, la instalación, sus pros y sus contras y saber si puede ser implementado en el proyecto de desarrollo que se está levando a cabo.

## Instalación

Para poder instalar MobSF en los diferentes entornos operativos, primero necesitamos descargar ciertos programas que son requerimientos para el funcionamiento de la aplicación.

Requerimientos para el análisis estático:

* Empezamos instalando [Git](https://git-scm.com/download/win)
* Instalamos [Anaconda Python **3.7**](https://repo.anaconda.com/archive/Anaconda3-2020.02-Windows-x86_64.exe) en versiones superiores aún no tiene soporte
* Instalamos [JDK 8](https://www3.ntu.edu.sg/home/ehchua/programming/howto/JDK_Howto.html) o superior
* Agregamos o instalamos desde 0 [Microsoft Visual C++ Build Tools](https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=BuildTools&rel=16)
* Instalamos (solo si es necesario) [[4]](#footnote-4) [OpenSSL](https://slproweb.com/download/Win64OpenSSL-1_1_1d.exe)
* Por último, descargamos [wkhtmltopdf](https://wkhtmltopdf.org/downloads.html) [[5]](#footnote-5)

Requerimientos para el análisis dinámico

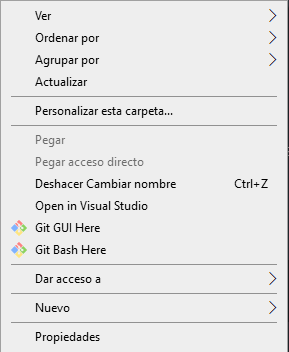
* Instalar [Genymotion](https://www.genymotion.com/fun-zone/) o [Android Studio Emulator](https://developer.android.com/studio) [[6]](#footnote-6)

Es importante que se instalen previamente a la descarga e instalación de MobSF.

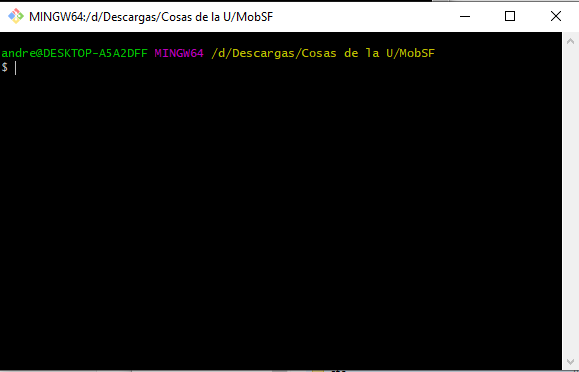
Una vez se tengan los requerimientos instalados, procederemos a instalar MobSF.

### Para Windows

Para empezar, primero debemos abrir Git en la carpeta en la que deseemos guardar nuestro programa, para esto solo necesitamos ir a esa carpeta deseada, darle click derecho donde veremos lo siguiente:



Le damos click en la opción de Git Bash Here y eso nos abrirá la siguiente consola:



Una vez aquí ingresamos el comando

* git clone <https://github.com/MobSF/Mobile-Security-Framework-MobSF.git>

Esto nos clonará el repositorio en la carpeta deseada.

Una vez terminado de clonar el repositorio en el cual se encuentra MobSF, en la misma terminal de Git Bash ejecutamos este comando:

* cd Mobile-Security-Framework-MobSF

Al entrar a la carpeta Mobile-Security-Framework-MobSF, ingresamos el siguiente comando:

* setup.bat

### Para Linux/Mac

De igual forma procedemos a abrir Git Bash en un fichero específico en el cual se almacenará nuestro programa y ejecutamos los siguientes códigos:

* git clone https://github.com/MobSF/Mobile-Security-Framework-MobSF.git
* cd Mobile-Security-Framework-MobSF
* ./setup.sh

Nota: Puede ser ejecutado también desde la terminal del sistema operativo, pero es mejor hacerlo en la de Git

Una vez terminado el proceso, nuestro programa ya estará instalado y podrá ser utilizado sin problema.

## Ejecutar MobSF

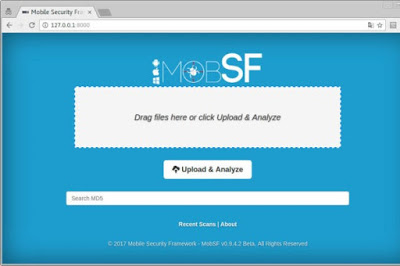
Para ejecutar MobSF en Linux o Mac, basta con colocar ./run.sh en la terminal del sistema o en la terminal de Git.

Para usar MobSF nos dirigimos al navegador de preferencia y colocamos http://localhost:8000/ y con esto ya podremos acceder a la interfaz gráfica que tiene MobSF.

Para ejecutar MobSF en Windows, basta con colocar run.bat en la terminal del sistema (cmd) o en la terminal de Git.

De igual forma que en Linux y en Mac, accedemos al navegador de preferencia o que predeterminado y colocamos http://localhost:8000/ para acceder a la interfaz gráfica de MobSF.

Veremos algo así:



Nota: Esta es la interfaz gráfica de versiones anteriores a la que estamos descargando.

## Funcionamiento

MobSF o Mobile Security Framework es una aplicación automática encargada de realizar pruebas de pen-test[[7]](#footnote-7), análisis de malware[[8]](#footnote-8) y hacer evaluaciones de seguridad de forma estática y dinámica.

### Análisis estático

Para realizar un análisis estático de una aplicación Android o iOS solo es necesario arrastrar la aplicación a su interfaz web, este buscará posibles problemas y mostrará los resultados cuando haya terminado.

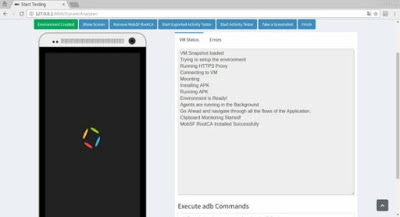
Luego podremos fácilmente verificar el resultado en diferentes páginas web que se encargan de clasificar los virus (Nessus analytics, VirusTotal, etc) para saber a qué nos enfrentamos

### Análisis dinámico

Para realizar un análisis dinámico es necesario tener una máquina virtual o Virtual machine que actuará como un HTTPS proxy [[9]](#footnote-9) (Esta máquina virtual fue previamente descargada en el momento de la instalación de requisitos)

Solo basta con hacer click en la opción “Start Dynamic Analysis” y este procederá a arrancar la máquina virtual si es que no la tenemos.

Luego daremos click en el botón “Create Enviroment” el cual nos instalará la aplicación que queremos analizar en la máquina virtual



Nota: Esta es la interfaz gráfica de versiones anteriores a la que estamos descargando.

En este punto tendremos que interactuar con la aplicación que se encuentra en la máquina virtual para que el Dynamic analyzer pueda inspeccionar cada instancia de esta, es bueno ser riguroso en este paso para así obtener un resultado mas detallado de la situación en la que se encuentra nuestra aplicación.

Una vez hayamos terminado de hacer eso, le damos click al botón “Finish” y aquí el Dynamic analyzer se tomará un tiempo para arrojar un informe con el resultado.

(Nota: Los informes pueden ser descargados)

## 

## Ventajas y Desventajas

### Ventajas

* Es efectivo a la hora de analizar.
* Informes rigurosos a cerca de los problemas en la aplicación.
* Entorno gráfico amigable para el usuario.
* Tiene una gran base de datos de virus.
* Detecta si el dispositivo puede ser usado para ataques de tipo ransomware[[10]](#footnote-10)

### Desventajas

* Requiere del uso de internet.
* Solo es una herramienta de análisis, no resuelve el problema
* Solo puede ejecutarse desde un entorno Windows, por lo cual en Linux o Mac es necesario implementarlo mediante un contenedor Docker o una máquina virtual.
* Se necesita de una computadora para analizar aplicaciones.

## Posibles soluciones

### Workbox Google Analytics

Es una aplicación pensada para optimizar la experiencia en la interacción usuario-aplicación offline [[11]](#footnote-11)

Implementar esta solución para solventar nuestro problema actual es una solución bastante acertada, pero con un problema de ejecución.

Para que esta aplicación funcione, también necesita internet; el motivo por el cual la incluyo como solución es porque es una librería de Google que puede ser implementada fácilmente en la aplicación simplemente agregando las siguientes líneas de instrucción:

* import \* as googleAnalytics from 'workbox-google-analytics';
* googleAnalytics.initialize();

Pero no sería implementado en la aplicación de estudiantes, si no en la aplicación de maestros teniendo en cuenta que según las encuestas todos ellos cuentan con conexión a internet en sus casas y esto funcionaría en el momento en que se conecte a una red de internet.

Esta solución también dejaría expuesto el RaspBerry Pi v4.

Para más información consultar:

* <https://developers.google.com/web/tools/workbox/modules/workbox-google-analytics>

## Conclusiones

1. No es posible implementar MobSF para este proyecto ya que para su correcto funcionamiento requiere tener una conexión a internet y un entorno Windows en cual ejecutarse.
2. Nos enfrentamos a un gran problema de implementación ya que todas las aplicaciones de seguridad de datos funcionan con conexión a internet; hay unas como la que mencioné anteriormente que deja las request[[12]](#footnote-12) en espera hasta el momento en que tenga conexión, pero no es lo óptimo.
3. En concordancia con el # anterior, la mejor solución sería construir un analizador propio, es de ir, desde 0, y manejar todo de forma local, son la dependencia de internet.

# Cibergrafía

* <https://www.boomernix.com/2018/06/analisis-de-malware-para-android-con.html>
* <https://mobsf.github.io/docs/#/>
* <https://databricks.com/glossary/what-is-unified-ai>
* <https://hackernoon.com/top-7-cross-platform-mobile-app-development-tools-and-frameworks-847i230qz>
* <https://mobsf.github.io/docs/#/requirements>
* <https://mobsf.github.io/docs/#/installation>

1. TIC: Tecnologías de la información y la comunicación [↑](#footnote-ref-1)
2. “EduApps” es un proyecto educativo que se quiere implementar en el Urabá Antioqueño para ayudar a los estudiantes y profesores de las instituciones educativas a mejorar la eficiencia de las clases desde un entorno TIC [↑](#footnote-ref-2)
3. Framework: Es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar [↑](#footnote-ref-3)
4. Digo solo de ser necesario porque es posible que ya lo tengan instalado. [↑](#footnote-ref-4)
5. Para instalar este programa, por favor seguir esta guía: <https://github.com/JazzCore/python-pdfkit/wiki/Installing-wkhtmltopdf> [↑](#footnote-ref-5)
6. Para configurar estos programas por favor seguir esta guía: <https://mobsf.github.io/docs/#/dynamic_analyzer> [↑](#footnote-ref-6)
7. Pen-test: Una prueba de penetración, o "pen-test", es un ataque a un sistema informático con la intención de encontrar las debilidades de seguridad y todo lo que podría tener acceso a ella, su funcionalidad y datos. [↑](#footnote-ref-7)
8. Malware: cualquier tipo de software que realiza acciones dañinas en un sistema informático de forma intencionada y sin el conocimiento del usuario. [↑](#footnote-ref-8)
9. Proxy: es un servidor —programa o dispositivo—, que hace de intermediario en las peticiones de recursos que realiza un cliente a otro servidor. [↑](#footnote-ref-9)
10. Ransomware: Ransomware o "secuestro de datos" en español, es un tipo de programa dañino que restringe el acceso a determinadas partes o archivos del sistema operativo infectado y pide un rescate a cambio de quitar esta restricción” [↑](#footnote-ref-10)
11. Offline: Fuera de línea / sin conexión a internet. [↑](#footnote-ref-11)
12. Request: En español “Petición” [↑](#footnote-ref-12)